UNIVERSITÉ DE
SHERBROOKE
Faculté de génie
Génie électrique et génie informatique

Conception électronique et informatique d'un robot mobile pour usage dans un environnement domiciliaire

Mémoire de maîtrise ès sciences appliquées
Spécialité : génie électrique

Richard CLOUTIER

Sherbrooke (Québec) Canada 01 2007

RÉSUMÉ

Un robot mobile permet d'étendre les sens et les capacités d'action des humains dans des environnements variés. Un de ces environnements est le domicile, où le robot peut agir pour rendre des services aux occupants directement ou à distance via une interface de communication. C'est dans ce cadre que le travail présenté dans ce mémoire se situe, et plus spécifiquement pour la conception électronique et informatique d'un prototype de base mobile téléopérée à distance dans une optique de prestation de soins à domicile en télésanté.

Aucune plate-forme de ce type existe sur le marché, et notre étude vise à mettre en évidence et répondre aux défis d'intégration sous-jacents à une telle conception. Ce mémoire vient étudier le choix des capteurs, leurs interfaces (énergétique et informatique) et leur disposition sur la plate-forme, la localisation de la plate-forme dans son environnement (par odométrie, balises infrarouges *NorthStar* et repérage laser), la planification de trajectoires, l'architecture logicielle ainsi que la prise de décision embarquée sur le robot versus sa téléopération à distance.

Nos résultats démontrent les performances du système conçu et ses fonctionnalités, servant de première preuve-de-concept pour une telle plate-forme. Le design électrique du robot ainsi que le choix de ses composantes nous permettent d'obtenir une autonomie énergétique de trois heures. L'ajout de capteurs de proximité permettent la mise en place d'un algorithme d'évitement d'obstacles où le robot ralentit sa vitesse à l'approche d'un obstacle. Le logiciel CARMEN est le système de localisation étudié qui donne la meilleure estimation au niveau de la position du robot dans l'environnement. Le robot est également capable de se déplacer de manière autonome dans l'environnement d'opération en utilisant une représentation de celui-ci qui a été faite au préalable. Des tests en laboratoire ont permis de valider le fonctionnement général du prototype tant au niveau de la prise de décision autonome que celui de l'exécution de commandes envoyées par un opérateur distant. Ce prototype s'avère donc être un excellent point de départ sur lequel des améliorations pourront être apportées quant à l'évitement d'obstacles et la reprise de trajectoire, et pour la tenue de tests dans de vrais domiciles.

REMERCIEMENTS

Je tiens à remercier mon directeur de recherche, M. François Michaud, pour son soutien tant au niveau académique que moral ainsi que pour la confiance qu'il m'a demontrée tout au long du projet. Ce support fut très apprécié.

Je tiens également à remercier mon co-directeur Michel Lauria ainsi que mes confrères qui ont travaillés sur le projet TéléRobot, Daniel Labonté et Marc-André Roux, pour leur assistance et leur soutien. Je veux aussi remercier les autres membres du LABORIUS qui ont été impliqués de près ou de loin dans ce projet. Le support de Dominic Létourneau, Carle Côté, Clément Raïvesky et des membres de RoboMotio inc. ont été particulièrement importants pour la réussite de ce projet.

Un remerciement spécial doit être fait à Éric Marquis et Marise Viens qui ont toujours été présents pour m'épauler et me donner de judicieux conseils. Merci à mes parents qui m'ont toujours soutenus et encouragés tout au long de mes études et ce, même dans les moments difficiles. La confiance qu'ils m'ont donnée a été une source de motivation importante pour ma réussite. Enfin, un merci tout spécial à ma muse et copine Caroline qui, en plus de son soutien et sa compréhension, m'a permis de trouver l'inspiration et la motivation nécessaire pour réaliser ce projet. Cet ouvrage est dédié à ces personnes.

TABLE DES MATIÈRES

1 INTRODUCTION **1**

2 ROBOTS ET NAVIGATION DANS LES DOMICILES **3**

2.1 Robots existants pour le soutien de personnes à domicile 3

2.1.1 Robots utilisés avec des personnes âgées 3

2.1.2 Robots de téléprésence 7

2.1.3 Robot de service 9

2.1.4 Sommaire 10

2.2 Navigation autonome 13

2.2.1 Méthodes de localisation 13

2.2.2 Méthodes de cartographie de l'environnement et génération de trajectoire 16

2.2.3 Détection et évitement d'obstacles 19

3 CONCEPTION ÉLECTRONIQUE **21**

3.1 Alimentation électrique 24

3.2 Interface avec les capteurs 27

4 LOCALISATION, PLANIFICATION ET SUIVI DE TRAJECTOIRES **33**

4.1 Localisation 33

4.1.1 Localisateur CARMEN 34

4.1.2 Odométrie 34

4.1.3 Système de balises infrarouges *NorthStar* 40

4.2 Planification de trajectoire 43

4.3 Discussion 47

5 CONCEPTION LOGICIELLE POUR LA PRISE DE DÉCISION 49

5.1 Architecture décisionnelle 50

5.1.1 Module Comportements 51

5.1.2 Module Exécutif 53

5.2 Architecture logicielle 55

5.3 Performances du système 59

5.3.1 Temps de réaction de l'architecture logicielle 59

5.3.2 Preuves de fonctionnement des comportements moteurs 61

5.3.3 Test de validation du suivi de trajet autonome 65

BIBLIOGRAPHIE 71

LISTE DES FIGURES

Figure 2.1 Robot Nursebot en action 4
Figure 2.2 Care-O-Bot II 5
Figure 2.3 Robotdoc 6
Figure 2.4 Wakamaru 6
Figure 2.5 ApriAlpha 7
Figure 2.6 Secur-O-Bot 8
Figure 2.7 Garcia 9
Figure 2.8 Coworker 9
Figure 2.9 Robot Roomba (images tirées du manuel d'utilisation du robot) . . . 10
Figure 2.10 Schéma de fonctionnement du système *NorthStar* 14
Figure 2.11 Différences au niveau des cartes 17
Figure 2.12 Exemple d'une carte CARMEN 18
Figure 2.13 Cartographie géométrique 18
Figure 2.14 Problématique de la table 20

Figure 3.1 Robots utilisés pour les tests à domicile. 22
Figure 3.2 Base du Télérobot assemblée 23
Figure 3.3 Schéma de câblage pour l'alimentation 26
Figure 3.4 Alimentation des différents modules ajoutés sur la base de propulsion 27
Figure 3.5 Liens de communication entre les différents périphériques 28
Figure 3.6 Position des capteurs infrarouges sur le robot 29
Figure 3.7 Réponse du sonar SRF10 30
Figure 3.8 Zones de détection offertes par les sonars 31

Figure 4.1 Zone couverte par le *NorthStar* (le graphique est inversé verticalement) 41

Figure 4.2 Division en cellule de l'environnement d'expérimentation 45
Figure 4.3 Arbre d'adjacence pour l'environnement d'expérimentation 46

Figure 5.1 Exemple d'architecture appliqué à un problème de navigation autonome 50
Figure 5.2 Architecture décisionnelle utilisée sur Télérobot 51
Figure 5.3 Implémentation du module Comportements 53
Figure 5.4 Vitesse maximale permise en fonction de la proximité des obstacles . 54
Figure 5.5 Schéma-bloc du module exécutif 55
Figure 5.6 Implémentation sous MARIE de l'architecture décisionnelle de Télérobot 57
Figure 5.7 Implémentation du laser et CARMEN 58
Figure 5.8 Déplacements de la plate-forme sur la planche 63
Figure 5.9 Position du Télérobot par rapport à la trajectoire optimale 64
Figure 5.10 Environnement de test simplifié 65
Figure 5.11 Exemples de suivi de trajectoires en mode autonome 67

LISTE DES TABLEAUX

Tableau 2.1 Résumé des différentes caractéristiques informatiques des systèmes embarqués 12

Tableau 3.1 Consommation énergétique des différents systèmes 24

Tableau 4.1 Exactitude de l'odométrie en ligne droite pour les robots sur des distances différentes 37

Tableau 4.2 Erreur sur l'estimation odométrique après une révolution (°) 38

Tableau 4.3 Erreurs à 250 mm/sec et 28°/sec (sans élastique) 39

Tableau 4.4 Erreurs à 250 mm/sec et 28°/sec (avec élastique) 40

Tableau 4.5 Erreurs à 100 mm/sec et 12°/sec (avec élastique) 41

Tableau 4.6 Spécifications techniques du système *NorthStar* 42

Tableau 4.7 Intervalles des erreurs du *NorthStar* en laboratoire 42

Tableau 4.8 Comparatif du *NorthStar* et de CARMEN au point de référence . . . 43

Tableau 4.9 Comparatif du *NorthStar* et de CARMEN en un point statique . . . 44

Tableau 4.10 Temps de recherche de trajets selon le mode en msec 47

Tableau 5.1 Correspondance entre l'architecture décisionnelle et l'architecture logicielle 58

Tableau 5.2 Temps de réaction suite à la réception d'une commande d'arrêt d'urgence (valeurs en msec) 59

Tableau 5.3 Temps de réaction pour un changement de vitesse (valeurs en msec) 60

Tableau 5.4 Temps de réaction suite à une demande de trajet autonome 60

Tableau 5.5 Temps de réaction pour l'arrêt complet à partir de deux vitesses initiales (valeurs en msec) 61

Tableau 5.6 Comparaison des distances d'arrêt pour le comportement *Avoid Front* 62

Tableau 5.7 Distance entre le point d'arrêt du robot et l'objectif 63
Tableau 5.8 Fichier de définition utilisé pour la preuve de fonctionnement 66

CHAPITRE 1

INTRODUCTION

Le phénomène du vieillissement de la population est présent dans la plupart des pays occidentaux. Ce phénomène amène une charge supplémentaire au niveau des différents réseaux de santé. Une solution envisagée pour diminuer cette surcharge est de permettre aux personnes âgées de rester le plus longtemps possible dans leur domicile. Il faut donc développer des méthodes qui permettent de les assister dans ce type de milieu.

Le but général du projet est de concevoir le prototype d'une plate-forme robotique, nommée Télérobot, adaptée pour opérer dans un environnement domiciliaire. Cette plate-forme permettra à un opérateur distant de voir, d'entendre et de se déplacer un peu partout dans le domicile du patient. C'est ce que nous appelons la téléprésence. Dans le cadre de ce projet, l'opérateur pourra être un clinicien, un thérapeute ou tout autre expert en soins de santé, ainsi qu'un aidant naturel. Le patient est une personne âgée nécessitant la présence de ces différentes personnes via le robot.

L'envergure d'un tel projet est importante car il aborde plusieurs aspects tels que la conception mécanique, le mécanisme de locomotion, l'électronique, la navigation autonome, la prise de décision, la telecommunication, l'interface opérateur et l'évaluation intégrée du système dans son ensemble. L'objectif principal du présent travail est de réaliser qu'une seule partie de ces points, soit la conception électronique et informatique de la plate-forme robotique. Plus spécifiquement, le robot doit pouvoir naviguer de façon autonome et selon les instructions d'un opérateur. Pour y arriver, le robot doit être en mesure d'éviter les obstacles de toutes sortes (e.g., escaliers, occupants, table, chaises, fauteuil, animaux, chaussures) qui sont présents dans l'environnement, et ce de jour autant que de nuit. Le robot doit aussi être en mesure de se positionner à des endroits précis dans le domicile (e.g., pièce, station de recharge) car aucune assistance de la personne âgée est considérée pour le bon fonctionnement du robot. Il doit enfin posséder les fonctionnalités audio et vidéo pour permettre à un

opérateur distant d'interagir avec le patient via la plate-forme robotique. Le défi principal de ce projet est donc d'intégrer différentes technologies pour réaliser l'ensemble des capacités demandées sur un seul et même système.

Le document se divise en cinq sections. Le chapitre 2 présente une revue des différentes technologies qui pourraient être utiles pour la réalisation de ce projet. Le chapitre 3 traite des choix qui ont été faits au niveau de la conception électronique du prototype. Le chapitre 4 présente les différents concepts liés à la navigation autonome ainsi que de son implémentation dans le logiciel de contrôle. Des résultats de tests de performance sur les différents systèmes de localisation sont présentés dans ce chapitre. Le chapitre 5 traite du fonctionnement et de la mise en œuvre du logiciel de contrôle haut-niveau.

CHAPITRE 2

ROBOTS ET NAVIGATION DANS LES DOMICILES

Cette section présente différents systèmes et approches en lien avec notre projet. La section 2.1 touche les robots existants à partir desquels il est possible de s'inspirer pour faire le développement de la nouvelle plate-forme. La section 2.2 passe en revue l'ensemble des points liés à la problématique de la navigation autonome en robotique mobile.

2.1 Robots existants pour le soutien de personnes à domicile

Les projets robotiques suivants ont été développés dans le but de répondre aux besoins de soutien des personnes à domicile. Chacun des robots présentés possède au moins une caractéristique qui se rapproche de notre projet. L'analyse se divise en trois sections. La première décrit des robots qui interagissent avec des personnes âgées ou des patients dans les hôpitaux. La deuxième touche les robots de téléprésence. La troisième présente un robot de service à domicile.

2.1.1 Robots utilisés avec des personnes âgées

Cinq robots font partie de cette catégorie.

1. **Nursebot**. Ce robot est conçu pour opérer dans une résidence pour personnes âgées [20] [22] (figure 2.1). Le robot sert d'escorte pour diriger les patients vers des endroits stratégiques (e.g., toilettes). Il opère dans une zone limitée de l'environnement pour une question de sécurité. Un système audio est aussi implémenté sur le robot pour les interactions entre l'opérateur et le patient. Un module d'aide cognitive (e.g., rappel

Figure 2.1 – Robot Nursebot en action

pour prendre un médicament) a été implémenté pour palier aux problèmes de perte de mémoire chez les personnes âgées. Un système de reconnaissance vocale est également utilisé pour commander le robot. Le robot pèse 34 kg et a une hauteur de 1210 mm. Il faut brancher le robot pour permettre la recharge des batteries. La plate-forme est dotée d'un balayeur laser SICK PLS ainsi que de 16 sonars. Ces senseurs sont utilisés par le système de navigation autonome du robot.

2. **Care-O-Bot II**. Ce système, montré à la figure 2.2, regroupe trois catégories de fonctionnalités [12]. Deux d'entre-elles sont partagées par notre projet ainsi que celui du Nursebot, soient l'aide à la mobilité et l'aide cognitive. La troisième est l'assistance aux personnes dans des tâches domestiques. Le robot possède un bras robotique pour déplacer des objets à la place du patient. La plate-forme de base est le MP-M470 de la compagnie Neobotix[1]. Ce robot est de grande dimension (470 mm × 470 mm × 756 mm) et son poids est de 150 kg. L'alimentation est fournie par 8 batteries de 48 V, pour une capacité totale de 84 Ah. Il faut brancher manuellement le robot pour le recharger. Le contrôleur principal PentiumIII 800 MHz est embarqué sur le robot et un

[1] www.neobotix.de

Figure 2.2 – Care-O-Bot II

écran plat de 15 pouces sert pour l'interface-usager. Un détecteur laser SICK LMS 200 ainsi que des interrupteurs de contact sont utilisés pour la détection d'obstacles. La communication entre les différents modules embarqués (e.g., contrôle du bras, contrôle du cou, etc.) est faite par un bus CAN. Un système de communication Ethernet sans fil peut être utilisé pour interagir avec le robot. Un comportement de navigation autonome permet au robot de suivre des trajets pré-établis dans l'environnement.

3. **Robot-Doc**[2]. Cette plate-forme est conçue dans le but d'être téléopérée par un médecin (figure 2.3) afin de faire des visites à distance dans un autre centre hospitalier. Ce robot utilise un écran d'ordinateur à la place de sa tête pour afficher le visage de l'opérateur. Le lien de communication Ethernet sans fil sert à l'échange bidirectionnel de flux audio/vidéo et des commandes de navigation. Le robot a une hauteur totale de 1700 mm et pèse 91 kg. La détection d'obstacles est faite par 24 senseurs infrarouges positionnés sur la base. Le robot n'a pas de comportement pour la navigation ou la recharge autonome.

4. **Wakamaru**[3]. Ce robot a été développé par Mitsubishi avec précisément comme but l'interaction avec des personnes âgées (figure 2.4). Un lien de communication Ethernet sans fil est utilisé pour téléopérer le robot. Wakamaru mesure un mètre et pèse 30 kg.

[2] www.Intouch-Health.com
[3] www.sdia.or.jp/mhikobe-e/products/etc/robot.html

Figure 2.3 – Robotdoc

Figure 2.4 – Wakamaru

Il a une autonomie de deux heures et se dirige automatiquement vers sa station de recharge. Le coût d'achat de cette plate-forme est de 14,250$ US. Ce robot est capable de se repérer dans un domicile en analysant les images prises au plafond. Il peut opérer de façon autonome pour reproduire des trajets établis à l'avance. L'identification et l'évitement d'obstacle est fait à l'aide de sonars et de senseurs infrarouges. Plusieurs processeurs sont embarqués sur le robot pour contrôler les interfaces vocales et le traitement d'images. La communication entre les modules est fait par une connection série. Sa tolérance quant à la dénivelation maximale du plancher est de 1 cm.

5. **ApriAlpha**. Ce robot a été développé par la compagnie Toshiba pour opérer dans un environnement domiciliaire (figure 2.5). Le robot communique via un lien Ethernet

Figure 2.5 – ApriAlpha

sans fil, Bluetooth ou par infrarouge. Un système audio bidirectionnel est aussi disponible. L'aspect intéressant est sa petite taille 350 mm × 350 mm × 380 mm et son poids de 10 kg. Il a une autonomie d'environ deux heures et la recharge n'est pas autonome. Le système utilise six sonars pour l'évitement d'obstacles. Les caméras vidéo sont utilisées pour permettre à un algorithme de vision d'identifier la présence d'une personne dans l'image. L'algorithme recherche la présence de vêtements en utilisant les caractéristiques de couleur et de texture de ceux-ci. Une fois qu'une personne est trouvée, il est possible de la suivre lors de ces déplacements tout en restant à une distance sécuritaire.

2.1.2 Robots de téléprésence

Par définition, la téléprésence permet à un opérateur de voir, d'entendre et de se déplacer un peu partout et même agir dans un environnement distant. Les caractéristiques de trois plate-formes robotiques de téléprésence sont présentées ici.

1. **Secur-O-Bot**. Ce robot montré à la figure 2.6 est une variante de la plate-forme Care-O-Bot. Elle est axée sur la télésurveillance en milieu industriel. Le système de base mesure 400 mm × 400 mm × 375 mm et pèse 30 kg. Il a une autonomie de huit heures. Les batteries sont de 24 V et ont une capacité de 40 Ah. La recharge du robot s'effectue en branchant manuellement le robot à un bloc de recharge. La caméra se trouve à une hauteur d'environ 1 mètre. Le contrôleur principal PentiumIII 800MHz est embarqué sur le robot. Le robot utilise un laser SICK LMS200 pour la détection et l'évitement d'obstacles. La communication entre les modules embarqués est faite via un

Figure 2.6 – Secur-O-Bot

bus CAN. Ce système est capable de mémoriser des trajets et ensuite être en mesure de les répéter de façon autonome. Le robot comprend aussi un système de téléopération. Les différentes commandes sont envoyées au robot en utilisant un lien Ethernet sans fil.

2. **Garcia**. Cette plate-forme, montrée à la figure 2.7, a été développée par la compagnie Acroname. La base a une dimension de 280 mm × 195 mm × 94 mm. Il est possible d'ajouter un mât de 406 mm, contrôlé en rotation et en inclinaison. Ce mât permet de donner une certaine flexibilité pour l'utilisation d'une caméra embarquée. Le robot est alimenté par six cellules de 7.2 V pour une capacité totale de 3 Ah. Le système de recharge est externe au robot. Il est possible d'utiliser un processeur XScale de 400 MHz ou de monter un système de communication par radio fréquence pour permettre un contrôle à distance du robot. Des sonars et des détecteurs infrarouge sont utilisés pour la détection d'obstacles.

3. **CoWorker**. Ce robot, montré à la figure 2.8, a été conçu spécifiquement pour la téléprésence. L'opérateur contrôle le robot via une interface web et un lien Ethernet sans fil. Les dimensions de cette plate-forme sont de 687 × 432 × 914 mm et elle pèse environ 34 kg. L'ordinateur de bord est un PentiumIII 800 MHz. Elle est munie d'une caméra, d'un sonar rotatif et de capteurs infrarouges. Ce systême n'a pas de comportement de recharge ou de navigation autonome.

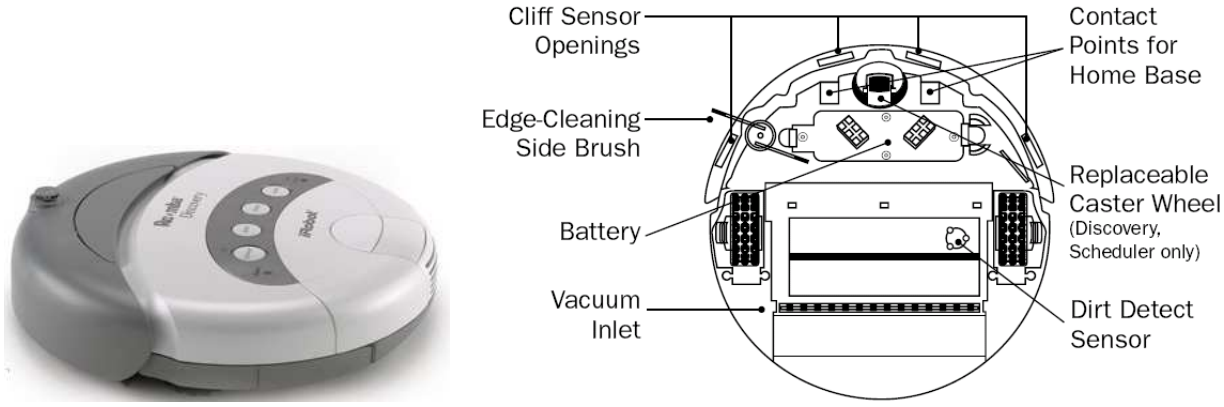
Figure 2.7 – Garcia

Figure 2.8 – Coworker

2.1.3 Robot de service

Il existe sur le marché différentes plate-formes dans cette catégorie. Les applications principales sont des aspirateurs ou des tondeuses à gazon. Leur fonction principale est de couvrir une surface de travail, de dimension variable, de manière autonome. Le robot Roomba, le premier et jusqu'à présent le seul succès commercial de robots de service à domicile (vendu à plus de 2 millions d'exemplaires partout dans le monde) résume bien les caractéristiques de fonctionnement de ce type de robot.

1. **Roomba**. Ce robot aspirateur pour les domiciles, illustré à la figure 2.9(a), couvre de manière autonome une surface qui peut être de dimension variable. Il est aussi pos-



(a) (b)

Figure 2.9 – Robot Roomba (images tirées du manuel d'utilisation du robot)

sible de le commander à l'aide d'une télécommande. Les nouvelles versions du robot permettent de programmer à l'avance le démarrage de la tâche de nettoyage. L'alimentation du robot est basée sur une batterie NiMH de 12 V, ce qui donne une autonomie de 2 heures en mode de fonctionnement normal. Roomba est en mesure de se recharger de manière autonome en autant que sa station de recharge se retrouve dans les limites de la surface de travail. Ces capacités de navigation autonomes sont limitées à se promener de manière aléatoire à l'intérieur de la surface de travail. Il mesure 350 mm de diamètre, 100 mm de haut et il a un poids de 3 kg. Une caractéristique intéressante de ce système est qu'il utilise des capteurs infrarouges pour détecter la présence d'un escalier. La figure 2.9(b) montre la position des différents capteurs. Ceux-ci pointent directement vers le sol et lorsque la distance retournée par ces capteurs est trop grande, le robot identifie une situation de faute. Des interrupteurs de contacts sont également utilisés pour identifier la présence d'obstacles (partie grise foncée à gauche de la figure 2.9(a))[4].

2.1.4 Sommaire

Le tableau 2.1 résume les différentes caractéristiques des robots présentés. Les méthodes de contrôle (téléopération ou navigation autonome) sont les points principaux à considérés. Ces

[4] www.irobot.com/images/consumer/cs/Roomba-owners-manual.pdf

informations sont utiles pour mettre en évidence le niveau technologique des plate-formes.

TABLEAU 2.1 – Résumé des différentes caractéristiques informatiques des systèmes embarqués

Robots	**Ordinateur**	**Senseurs**	**Navigation autonome**	**Recharge autonome**	**Dimensions** (mm)	**Poids** (kg)	**Autonomie** (heures)	**Prix** $US
Nursebot	ND	Laser Sonars			h : 1210	34	ND	ND
Care-O-BotII	PIII 800 MHz	Laser Sonars Interrupteurs	X	X	L : 470 l : 470 h : 756	150	ND	ND
Robotdoc	ND	Infrarouges			h : 1700	91	ND	ND
Wakamaru	ND	Sonars Infrarouges			h : 1000	30	ND	14250
ApriAlpha	ND	Sonars			L : 350 l : 350 h : 380	10	2	ND
Secur-O-Bot	PIII 800 MHz	Laser	X	X	L : 400 l : 400 h : 1000	30	8	ND
Garcia	Xscale 400 MHz	Sonars Infrarouges			L : 280 l : 195 h : 500	ND	ND	2000
CoWorker	PIII 800 MHz	Sonars Infrarouges			L : 682 l : 437 h : 914	34	5	50000
Roomba	ND	Infrarouges Interrupteurs		X	d : 350 h : 100	3	2	370

Légende : h = hauteur : L = longueur : l = largeur : d = diamètre ND = information non-disponible

2.2 Navigation autonome

Comme le montre le tableau 2.1, peu de plate-formes robotique arrivent à naviguer de manière autonome. Une telle capacité demande au robot d'être en mesure de se localiser dans l'environnement d'opération et de naviguer avec l'aide de cette représentation des lieux tout en évitant les obstacles se présentant sur son chemin.

2.2.1 Méthodes de localisation

Les méthodes de localisation peuvent être divisées en trois catégories [10] : relative, absolue et mixte.

Les méthodes de localisation relative consistent à prendre en considération l'ensemble des déplacements effectués par la plate-forme par rapport à un point de départ. Aucune référence à l'environnement extérieur est utilisée pour la localisation. Une technique possible est d'utiliser des encodeurs de roues pour mesurer les déplacements effectués par odométrie. Une difficulté est qu'une compensation doit être effectuée lors des rotations de la plate-forme, car la roue extérieure tourne alors plus que la roue intérieure. Les roues peuvent aussi glisser, ce qui vient fausser les lectures. Une autre technique consiste à utiliser des capteurs inertiels (accéléromètre, gyroscope) [5], et faire l'intégration des différentes valeurs lues pour évaluer le déplacement effectué. Les deux techniques sont vulnérables aux erreurs de précision présentes sur chaque lecture des capteurs. Ces erreurs sont cumulées dans le temps, ce qui rend difficile ou même impossible d'utiliser seulement ces techniques pour dériver la position du robot.

Les méthodes de localisation absolue utilisent l'information provenant de balises ou de points de repères ayant des coordonnées connues et fixes. Il est donc possible de calculer la position à partir des distances mesurées entre le robot et les balises. Différentes méthodes peuvent être appliquées :

– Balises ultrasoniques [21] [3]. L'approche consiste à utiliser des balises ultrasoniques équipées d'un système de communication par radio-fréquences. Les balises sont à des positions connues et fixes. À tour de rôle, un minimum de trois émetteurs ultrasoniques envoient une onde ultrasonique. La synchronisation des différents systèmes est fait par le lien radio. Le

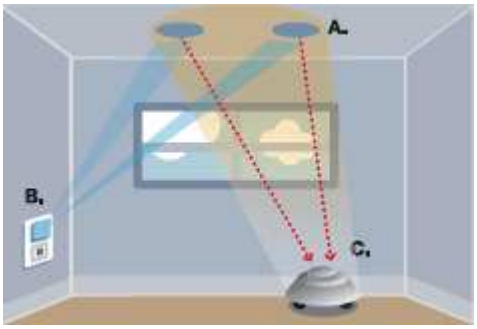


Figure 2.10 – Schéma de fonctionnement du système *NorthStar*

robot évalue avec l'aide d'un ou plusieurs récepteurs ultrasoniques le temps de réception du signal. En utilisant la vitesse du son comme constante, il est possible de déduire la distance entre le récepteur et les sources avec l'aide d'un algorithme de triangulation [2].

– Balises infrarouges. Le système *NorthStar* de la compagnie Evolution Robotics utilise des émetteurs infrarouges qui envoient des signaux vers le plafond de la pièce. Le récepteur, qui est positionné sur le robot, évalue sa position en fonction de la puissance ainsi que par l'orientation des signaux reçus. La figure 2.10 présente le principe de fonctionnement du système et cette explication est présentée en détail à la section 4.1.3.
– Repérage par signature laser. Un capteur laser de proximité comme le SICK LMS 200 couvre une zone de 180° avec une précision de 0,5°. Il fonctionne sur une distance de 30 m avec une résolution de 10 mm et le taux de rafraîchissement des lectures est de 75Hz[5]. Il permet d'obtenir une représentation précise en 2D des éléments structurels entourant le robot. Une telle information peut être utilisée pour situer le robot par rapport à une carte de l'environnement, localisant ainsi le robot de façon absolue par rapport à cette connaissance de l'environnement. Toutefois, certains matériaux comme les vitres et les miroirs peuvent engendrer certaines difficultés de perception. C'est aussi un capteur qui demande 17,6 W de puissance, générée à 24 Vdc. Il pèse 4,5 kg et coûte 8400 $.
– Repérage par signature de sonars. Similaire en principe au repérage par signature laser, cette technique utilise plutôt plusieurs sonars de proximité autour du robot pour obtenir la représentation 2D de l'environnement. Bien que les sonars sont moins coûteux et moins énergivores qu'un laser, ils sont aussi moins précis (l'ouverture du cône de détection

[5] www.sickusa.com

est de 30°) et ils sont très sensibles au bruit électrique et aux chocs. Le taux de rafraîchissement des lectures est de l'ordre de 25 Hz. Dépendamment du type de surface de l'objet et de l'angle d'incidence de l'onde ultrasonique, il est possible que l'onde émise ne soit pas réfléchie par la surface (e.g., tissu) ou qu'elle rebondisse sans revenir vers l'emetteur (phénomène connu sous le nom de réflexion spéculaire). Un capteur ultrasonique peut réagir à une onde envoyée par un autre capteur sur le robot (*crosstalk*) ou recevoir l'echo envoyé lors d'une lecture précédente (réflexion fantôme).

– Repérage par signature visuelle. Ce type de repérage tente de tirer des informations caractéristiques d'images prises dans l'environnement [24] [14] [19]. Par exemple, il est possible d'extraire le contour des objets et d'ensuite faire une comparaison avec des points connus de l'environnement (e.g., meuble, cadre de porte [24]). Une autre méthode consiste à trouver les variations importantes au niveau des couleurs dans l'image [24]. L'image est divisée en petites zones indépendantes, ce qui permet de générer une signature unique pour chaque image. L'image recueillie est faite sur le sens de la hauteur. Cette caractéristique est différente des méthodes par laser où par sonars, qui elles donnent une représentation sur un plan parallèle au plancher. Il est donc possible d'ajouter des éléments sur les murs pour ajouter des points de repères dans l'environnement. Un exemple est le robot Sage[6] de l'Université Carnegie Mellon [19]. Des marqueurs de couleurs sont disposés dans l'environnement à des positions connues par le robot. Ils sont positionnés près du plafond dans le but qu'ils soient facilement détectables. Il existe deux types de marqueurs. Les marqueurs 2D sont tout simplement des carrés de couleur rose disposés aux extrémités des corridors. Le robot peut évaluer son orientation par rapport à un marqueur selon la position de ce point de repère dans l'image vidéo. Pour les marqueurs 3D, un carré noir est placé à l'avant du rose. À partir de la différence de grosseur des deux carrés, il est possible d'évaluer la distance du robot par rapport au marqueur en plus d'avoir l'estimation au niveau de l'orientation.
– Méthodes de localisation dites mixte. Ces méthodes consistent à combiner la localisation relative et absolue. Par défaut, la localisation est faite de façon relative jusqu'à ce qu'une balise ou un point de repère, ayant une position connue, soit identifiée. La position du

[6] www.cs.cmu.edu/ illah/SAGE

robot peut alors être réinitialisée par rapport à des points absolus de l'environnement, diminuant les erreurs liées à la localisation relative dans la mesure où les repères sont uniques et précis. Outre les méthodes présentées précedemment, une approche consiste à d'utiliser la technologie RFID [13]. Plusieurs modules passifs RFID sont déployés à des endroits fixes et connus dans l'environnement, et les coordonnées des différents points sont enregistrées dans le robot. La localisation du robot est basée sur l'odométrie qui est recalibrée lors du passage du robot près d'un des modules RF (moins de 60 cm).

2.2.2 Méthodes de cartographie de l'environnement et génération de trajectoire

Une fois localisé, il est possible au robot de se situer dans l'environnement d'opération et de planifier un trajet pour se rendre d'un point A à un point B. Pour y arriver, le robot a besoin d'une carte de l'environnement. Cette carte peut être fournie préalablement au robot ou être construite dynamiquement, soit lors d'une étape de cartographie préalable à la mise en opération du robot, soit directement par les actions autonomes du robot. Cette dernière technique est qualifiée de SLAM (*Simultaneous Localization and Mapping*). Le SLAM est une problématique de recherche encore d'actualité qui demande de pouvoir compenser les erreurs de localisation du robot lors de la construction d'une carte de l'environnement [25]. Les approches utilisables actuellement cherchent plutôt à exploiter des cartes pré-établies.

Pour être en mesure de localiser le robot sur une carte, il faut choisir entre différentes approches illustrées à la figure 2.11. Les possibilités sont :

1. Métrique. Cette méthode consiste à diviser l'environnement en plusieurs cellules de dimensions identiques [10] [25]. Une valeur est attribuée à chaque case en fonction de la probabilité qu'un obstacle empêche le robot de se déplacer au centre de celle-ci. Dans le logiciel de navigation CARMEN [18], il est possible de visualiser ces valeurs avec des teintes de gris (blanc = aucun obstacle : noir = obstacle présent). La figure 2.12 illustre un exemple d'une carte CARMEN. Les zones grisées sur le côté droit de la représentation sont des zones d'incertitudes qui ont été identifiées lors de la génération

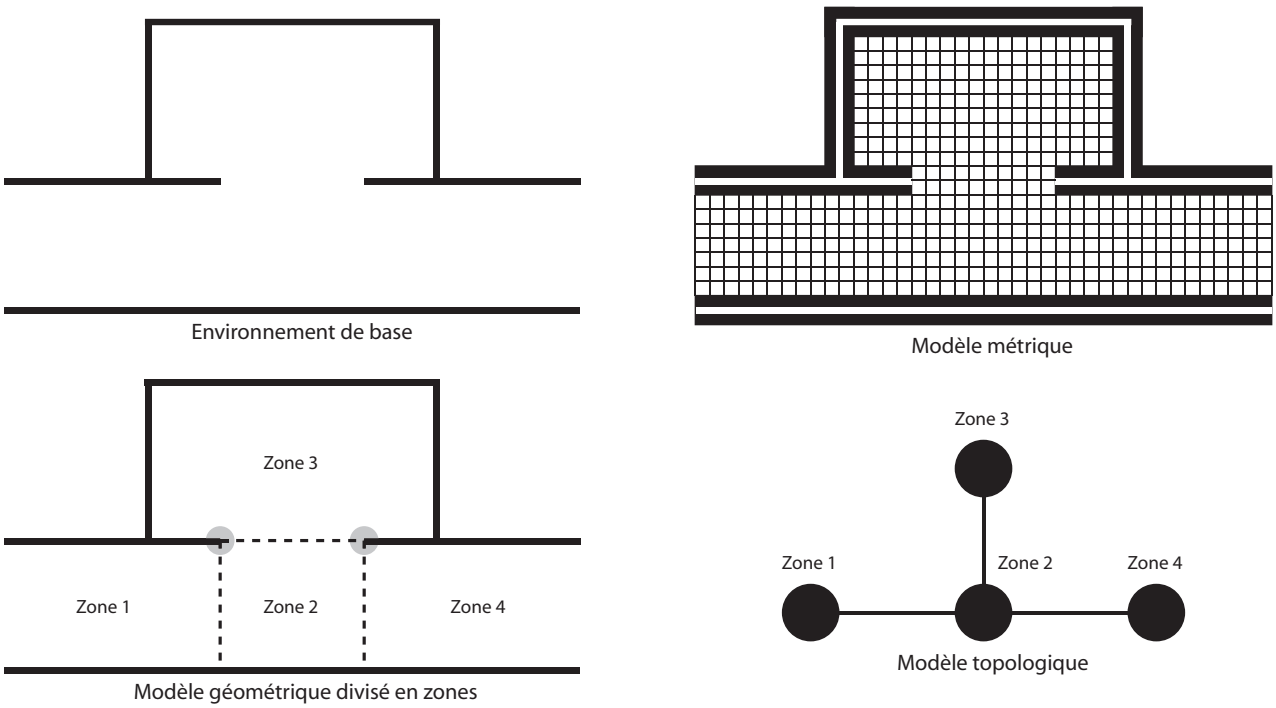


Figure 2.11 – Différences au niveau des cartes

de la carte. Les murs sont représentés par des lignes noires.

2. Géométrique. L'environnement est alors représenté par une série de formes géométriques simples comme des points ou des droites [10] [15]. La figure 2.13 est un exemple où les obstacles sont définis comme étant des lignes droites. Le localisateur prend en compte les lectures des capteurs de proximité et il essaie de retirer les formes géométriques qui sont présentes autour du robot. La position des différentes formes et la position de celles-ci les unes par rapport aux autres sont les critères pour identifier la position du robot. Le problème majeur est qu'il est parfois difficile ou même impossible de créer une carte précise dûe à la présence de formes trop complexes à modéliser. De plus, les différents obstacles sont définis par leur forme ainsi que par leur position dans l'environnement (coordonnées absolues).

3. Topologique. Des points de repères sont utilisés ici comme références pour identifier des lieux dans l'environnement, et un graphe d'adjacence vient représenter les accès possibles à ces lieux [10] [25] [24]. Il en résulte une représentation compacte mais limitée de l'environnement. Les points de repères peuvent être des éléments déjà présents ou

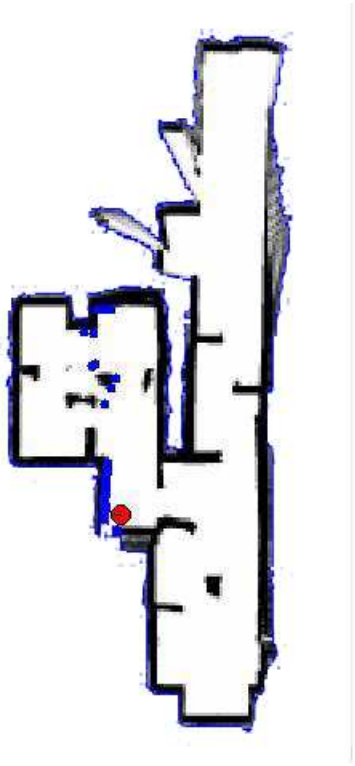


Figure 2.12 – Exemple d'une carte CARMEN

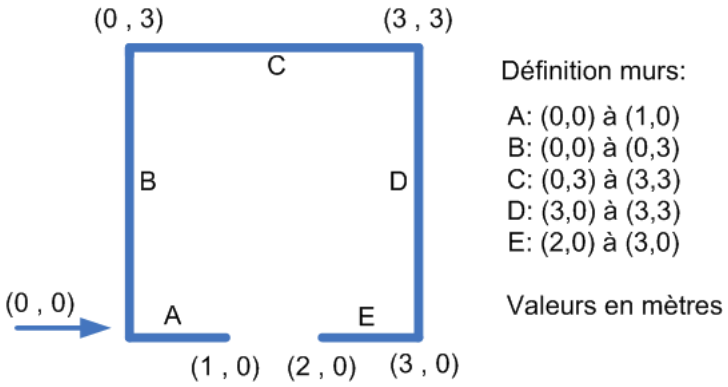


Figure 2.13 – Cartographie géométrique

des éléments ajoutés à l'environnement d'opération. Le point à considérer est que les repères doivent avoir une "signature" unique pour éviter de les confondre entre-eux. Dans la figure 2.11, les points de repères utilisés dans la représentation topologique sont les coins des cadres de porte.

Une fois positionné, il est possible de planifier la trajectoire à suivre en ayant une représentation de l'environnement. Par exemple, la section 4.2 explique le fonctionnement de l'algorithme utilisé pour la génération d'un trajet en fonction de la position présente du robot et de celle de l'objectif.

2.2.3 Détection et évitement d'obstacles

Puisque l'environnement domiciliaire est dynamique, il est possible qu'un élément de l'environnement soit déplacé par rapport à la carte ou qu'il ne soit pas identifiable sur celle-ci. Dans ce cas, le robot percuterait l'obstacle sans le savoir. Il faut donc ajouter des senseurs qui permettront au robot de détecter et d'éviter les obstacles.

La première chose à faire est de déterminer l'étendue de la zone de détection souhaitée afin de choisir les senseurs à utiliser. Comme mis en évidence au tableau 2.1, plusieurs types de senseurs comme un laser, des sonars ou des détecteurs de distance infrarouge sont couramment utilisés pour cette tâche. Le laser a l'avantage d'être très précis et il couvre une zone importante (e.g., 180° pour les laser SICK). Les sonars ont l'avantage de couvrir une zone plus large que les infrarouges qui détectent seulement en ligne droite. Il reste l'utilisation d'interrupteurs de contact mécaniques comme dernier niveau de détection. Étant donné les limitations sur chacun des capteurs (e.g., précision, exactitude, taux de rafraîchissement, couverture, détection de différents matériaux), il s'avère utile d'en combiner plusieurs pour assurer une détection plus fiable et robuste.

Ces limitations sont autant plus grandes dans les domiciles où les obstacles peuvent être de différentes formes (e.g., table avec pattes, meubles, animaux) sont présents. La figure 2.14 est un exemple qui peut se produire dans le domicile. Un autre cas important consiste à détecter la présence d'un escalier en percevant l'absence de plancher (voir figure 2.9(b)). Une solution

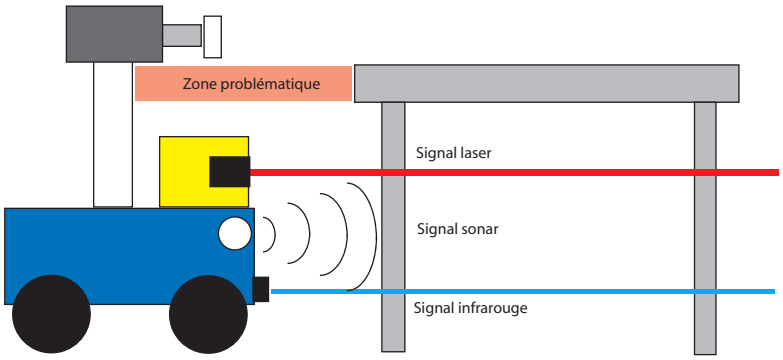


Figure 2.14 – Problématique de la table

consiste à utiliser des détecteurs de proximité infrarouge orientés vers le plancher. Une autre alternative est le concept du mur virtuel mis en place soit avec une bande réfléchissante sur le plancher [11] ou avec un émetteur infrarouge. Dans les deux cas, une limite “virtuelle” est créé et le robot ne doit pas la dépasser. Enfin, le cas où l’inclinaison du robot est trop importante (e.g., passage d’un seuil de porte) doit aussi être détecté afin d’éviter que le robot ne tombe sur le côté.

CHAPITRE 3

CONCEPTION ÉLECTRONIQUE

Le chapitre 2 illustre bien la diversité des considérations entourant la conception d'une plate-forme robotique mobile, ainsi que de l'absence de plate-formes répondant aux exigences d'une application de téléprésence dans les domiciles. En effet, les robots sont trop petits ou trop volumineux, sans capacités de téléprésence ou d'évitement d'obstacles tels que les escaliers ou les seuils de porte, et demandent des améliorations importantes pour les rendre utilisables dans les applications de téléprésence à domicile.

Pour confirmer cette observation, des essais préliminaires ont été réalisés à l'été 2004. Deux plate-formes robotiques, un CoWorker et un Magellan, ont été utilisées dans deux domiciles différents [4]. Les robots en question sont illustrés à la figure 3.1. Le robot CoWorker est doté d'une caméra couleur placée sur un mat mobile, de capteurs infrarouges de proximité, d'un sonar rotatif, d'un lien Ethernet sans-fil et d'une interface de téléopération par commande visuelle. Le robot Magellan était alors équipé d'une caméra couleur, d'un lien Ethernet sans fil, d'un laser SICK PLS et d'un système de cartographie métrique avec le navigateur CARMEN. Ces essais ont confirmés le besoin de développer une nouvelle plate-forme pour répondre aux exigences de l'environnement d'opération et de l'interface-opérateur. De plus, les groupes de discussion focalisée avec des cliniciens, des aidants naturels et des personnes âgées ont permis d'identifier le télégardiennage et la téléassistance dans la prestation de soins comme applications cibles de la plate-forme, affectant du même coup les spécifications pour le nouveau robot.

Parmi les observations faites lors de ces essais, les points suivants touchent directement la conception électronique du prototype de ce nouveau robot, baptisé Télérobot :

– Une propulsion de type différentielle (soit une roue de propulsion indépendante sur chaque côté du robot) permet des déplacements appropriés dans le domicile. Ce mode de propulsion est compact et permet d'utiliser des algorithmes de contrôle simples. Une propulsion

(a) CoWorker (b) Magellan

Figure 3.1 – Robots utilisés pour les tests à domicile.

de type différentielle est préférable à une propulsion de type omnidirectionnelle pour ne permettre que des déplacements dans la direction où l'image de la caméra est disponible.

– Un système de détection des obstacles est important car il est difficile à un opérateur de détecter la présence d'obstacle en utilisant seulement la vue vidéo de la caméra de téléopération comme référence. Utiliser un capteur laser est une solution adéquate mais coûteuse énergétiquement et en masse. L'utilisation d'infrarouge semble adéquate mais seulement à courte portée (15-20 cm), tandis que les sonars s'avèrent utiles pour des portées moyennes (15-75 cm). Il est également important pour le robot d'être en mesure de détecter la présence d'escaliers.
– Le robot doit être en mesure de naviguer d'un point A à un point B de façon autonome dans le domicile. La navigation par repérage laser est fonctionnelle, mais elle demande des ajustements au niveau de la cartographie.
– Il faut faire attention aux changements brusques de vitesse lorsque la caméra de téléopération se trouve haute par rapport à la base de la plate-forme. Des oscillations sont alors perceptibles dans le flux vidéo, ce qui est déplaisant pour le téléopérateur. De grandes oscillations sont aussi engendrées lorsque le robot passe au-dessus d'un seuil de porte. Une suspension appropriée doit être conçue.

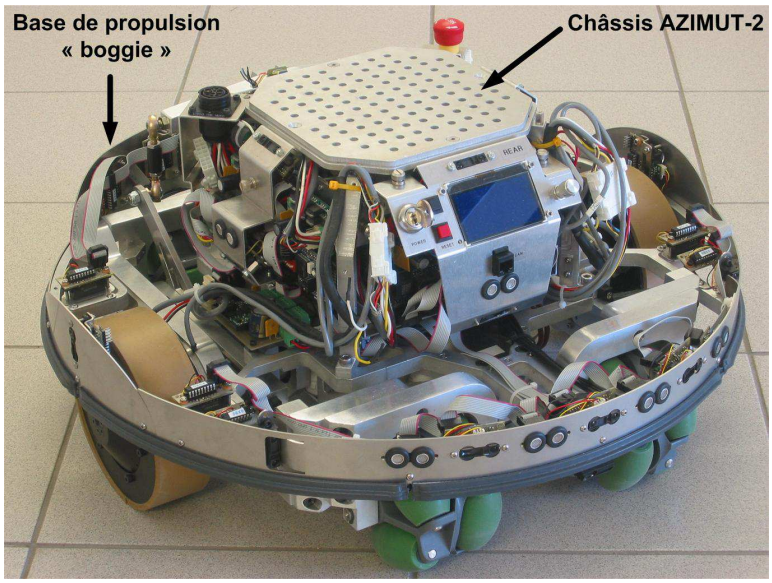


Figure 3.2 – Base du Télérobot assemblée

– Une autonomie énergétique de deux heures est souhaitable pour les usages du robot.

Le prototype du Télérobot résultant est illustré à la figure 3.2. Il est construit à même la plate-forme modulaire robotique AZIMUT-2 [16]. Il reprend deux roues de propulsion ainsi que le module de traitement et en ajoutant un nouveau système de suspension à “boggie” et des capteurs de proximité. Le module de traitement est constitué d'un ordinateur Pentium M 1,7 Ghz avec 512 Moctets de RAM, un disque dur de 60 Goctets et il fonctionne sous le système d'exploitation Linux. Cet ordinateur sert à la mise en œuvre du logiciel de contrôle de la plate-forme, et s'interface avec les différents sous-systèmes du robot (e.g., distribution de l'énergie, senseurs, actionneurs) via des contrôleurs dédiés. Ces contrôleurs sont des microcontrôleurs PIC18F de Microchip et interagissent entre eux par un bus de communication CAN.

Pour la plate-forme Télérobot, deux contrôleurs pour les roues de propulsion, déjà présents sur AZIMUT-2, sont utilisés. Une roue de propulsion est constituée d'un moteur installé à même la roue. Les encodeurs sont positionnés directement sur l'arbre d'entraînement du moteur. À l'interne du moteur-roue se trouve un système d'engrenage qui divise par quatre la vitesse de rotation de la roue. La précision des encodeurs est de 36000 pulses par tour. Avec un diamètre de roue de 161 mm, le résolution en terme de distance est de 0,014 mm. Une commande de vitesse envoyée au contrôleur d'un moteur de propulsion est exprimée

en termes de pulses/seconde, un pulse étant une unité pour l'encodeur de roue rattachée au moteur en question. En déplacement longitudinal, l'intervalle de vitesse disponible pour la plate-forme Télérobot est de 47 mm/sec à 1100 mm/sec. L'intervalle au niveau de la vitesse de rotation est de 5°/sec à 265°/sec.

Les autres éléments électroniques du Télérobot, soient son alimentation électrique et l'interface avec les capteurs, ont été conçus spécifiquement pour la plate-forme et sont décrits dans les prochaines sous-sections.

3.1 Alimentation électrique

La première étape pour la conception du système d'alimentation du Télérobot fut de mener une étude au niveau de la consommation énergétique des différents éléments à intégrer sur la plate-forme. Le tableau 3.1 résume les données utilisées impliquant différents systèmes sensoriels installés sur le robot.

TABLEAU 3.1 – Consommation énergétique des différents systèmes

Système	**Alimentation (Volt)**	**Puissance (Watt)**
Module AZIMUT-2 (moteurs à l'arrêt)	24	50
Module AZIMUT-2 (moteurs à 350mm/sec)	24	100
Senseurs (interface, 4 sonars, 16 IR)	24	5
Vidéophone (CareStation)	12	22
Caméra web (Sony RZ230)	12	13,2
Localisateur infrarouge (*NorthStar*)	12	0,66
Laser (SICK LMS200)	24	17,5

La puissance envisagée lorsque le robot est immobile est de 80,43 W et de 130,43 W en mouvement. La vitesse de 350 mm/sec est la vitesse maximale utilisée lors de la première phase de test du Télérobot. En prenant comme hypothèse que le robot est en mouvement 50% du temps, la consommation énergétique moyenne devrait être de 105,43 W. Des batteries de 24 Vdc doivent être en mesure de fournir en moyenne 4,4 A en continu pendant deux heures pour répondre à la spécification d'autonomie énergétique. Pour établir la puissance maximale requise, nous avons considérée la valeur maximale qui peut être consommée le

module AZIMUT-2. Le cas extrême est lorsque la plate-forme fait un changement brusque au niveau de la consigne de vitesse (de -1100 mm/sec à +1100 mm/sec). La puissance observée au niveau du module fut de 200 W, ce qui indique des courants de pointe de 8,3 A.

La figure 3.3 montre le schéma électrique du Télérobot, constitué du châssis d'AZIMUT-2 avec sa base de propulsion de type "boggie". Le modèle de pile VHD de la compagnie SAFT a été sélectionné. Ces piles sont de types NiMH avec une tension nominale de 1,2 Vdc et une capacité de 8,5 Ah. Elles sont similaires à celles utilisées sur AZIMUT-2. Les batteries sont configurées en deux paquets de 20 piles branchées en série (Batt1 et Batt2), ce qui donne une tension résultante de 24 Vdc pour chaque paquet. Avec une capacité énergétique de 17 Ah, il est théoriquement possible d'atteindre une autonomie de 3,87 heures, ce qui répond à la spécification. Cette évaluation est extrême dans le sens où tous les capteurs considérés au tableau 3.1 ne seront pas requis simultanément pour l'usage du Télérobot. Le temps de recharge des batteries est d'environ 3 heures. Les chargeurs Ch1 et Ch2 servent à recharger les batteries, tandis que la source d'alimentation 24 V permet d'alimenter le robot lorsque les batteries se rechargent. Il a fallu ajouter un branchement entre les deux cartes de gestion d'alimentation pour amener la tension de la source externe à la carte de gestion de Télérobot. La raison de ce choix est qu'il était difficile de faire une modification à la prise où se branche le câble de branchement sur AZIMUT-2.

Un point important à considérer dans la conception fut de s'assurer que le module d'alimentation alimente en tout temps et sans interruption les systèmes du robot, qu'il soit en recharge ou en opération. Le temps de réaction du système lors d'un branchement ou débranchement de l'alimentation externe doit être assez rapide. Un circuit électronique gère la sélection de la source d'alimentation qui est utilisée par le robot. La sélection des sources est représentée par des interrupteurs sur la figure 3.3. Les deux batteries sont sélectionnées en même temps lorsque le niveau de tension de la source externe est inférieur à 20 V. Dans le cas contraire, c'est l'alimentation externe qui est sélectionnée. Les batteries sont protégées par les disjoncteur D1 et D2 (de 20 A). Ces disjoncteurs servent également d'interrupteur pour débrancher les batteries du reste du système.

Le système de gestion d'alimentation énergétique d'AZIMUT-2 permet de lire le niveau de

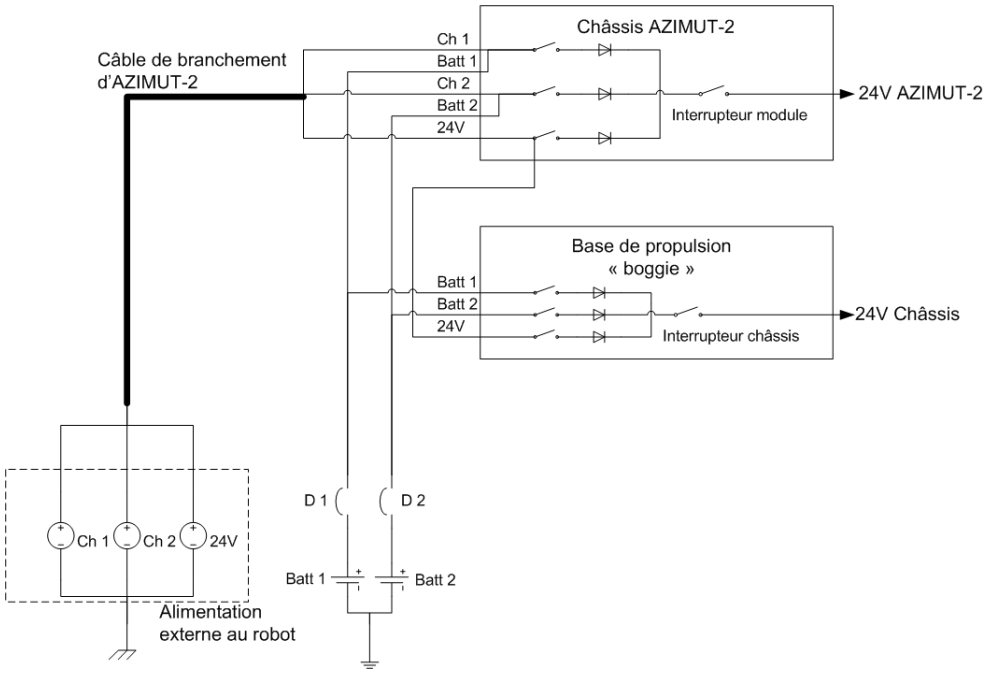


Figure 3.3 – Schéma de câblage pour l'alimentation

tension des batteries et de la source d'alimentation externe (précision 0,1 V), le niveau de puissance instantanée (précision au 0,1 W) et de courant (précision 1 mA). La valeur de puissance (en par conséquent celle de courant) est celle qui transite par le module d'alimentation d'AZIMUT-2, elle ne considère pas le courant qui transite par la base de propulsion. Toutefois, de nouveaux modules ont dû être développés pour les nouveaux éléments à intégrés au robot, soient les sonars et les infrarouges, le vidéophone, la caméra, le localisateur *NorthStar* et le laser. Un second système de gestion d'alimentation sert donc à alimenter ces nouveaux systèmes, avec comme objectif de pouvoir faire des tests avec les systèmes ajoutés sans avoir le module AZIMUT-2 monté sur la base de propulsion du Télérobot. Ce module est une copie conforme du module d'AZIMUT-2 sauf que les informations au niveau de la tension des batteries, de la puissance instantanée et du courant ne sont pas disponibles. La figure 3.4 montre le branchement des différents systèmes présents sur le Télérobot.

Les éléments suivants étaient actifs sur le robot lors de la première phase d'expérimentation :

1. Module AZIMUT-2,
2. Senseurs de proximité,

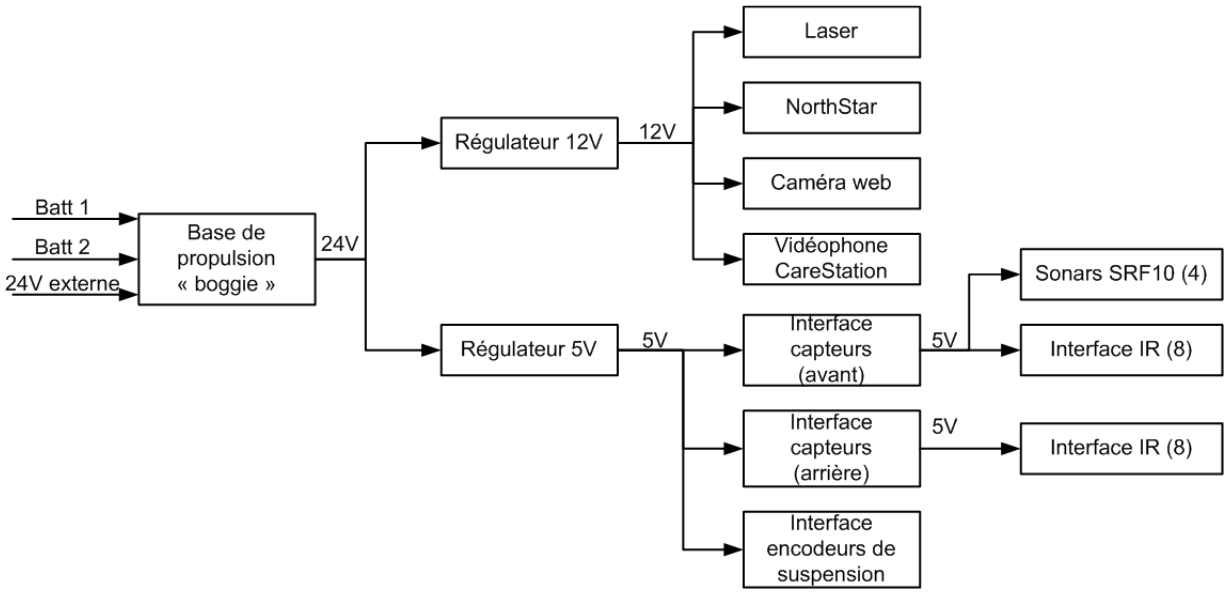


Figure 3.4 – Alimentation des différents modules ajoutés sur la base de propulsion

3. Localisateur infrarouge (*NorthStar*),
4. Capteur laser (SICK LMS200),

et l'autonomie observée fut de trois heures. Ce qui prouve que l'autonomie recherchée est atteinte pour des conditions d'opération normales.

3.2 Interface avec les capteurs

La plate-forme Télérobot doit être munie de capteurs permettant la localisation du robot et la détection des obstacles. La figure 3.5 illustre les interfaces entre les différents systèmes du Télérobot avec leurs liens de communication. Deux cartes d'interfaces pour les capteurs de proximité ont été développées pour diviser le temps d'acquisition et de traitement par un facteur deux. Les cartes d'interfaces développées échangent avec le module AZIMUT-2 via un bus de communication CAN. Un lien I2C est utilisé pour la communication entre les cartes d'interfaces des senseurs de proximité et les senseurs. Outre le localisateur *NorthStar* et le laser, il fallait positionner des sonars et des infrarouges pour la détection à courte portée. Afin d'établir la disposition, nous avons évalué les champs de perception propres à ces capteurs pris individuellement. Les tests consistaient à placer un obstacle fixe à l'avant

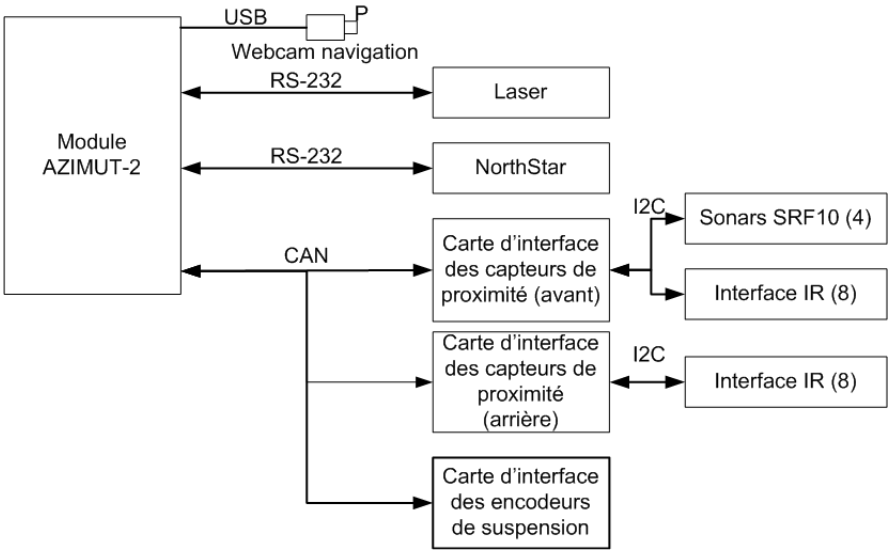


Figure 3.5 – Liens de communication entre les différents périphériques

du capteur et à comparer la valeur retournée avec la distance réelle de l'obstacle qui est mesurée manuellement .

Les performances des capteurs infrarouges (GP2D12 et GP2D120) respectent les spécifications du manufacturier au niveau des distances de détection. Pour le GP2D12, la distance de détection se situe entre 100 mm et 800 mm et elle est de 40 mm à 300 mm pour le GP2D120. Dans les deux cas, l'ouverture du cône de détection est d'environ 4°. La réponse du capteur est une tension analogique et elle n'est pas linéaire. Il faut mettre en place une table d'interpolation pour permettre d'obtenir la distance de l'obstacle. L'unité de mesure utilisée est le mm. La précision est de l'ordre de +/- 10 mm dans la première moitié de la zone de détection, et elle passe à environ +/- 20 mm dans la deuxième. Étant donné que les zones de détection des capteurs infrarouges ne se chevauchent pas, il est possible de faire une lecture simultanée de ceux-ci. Bien que les zones couvertes par ces capteurs ne permettent pas de faire un comportement d'évitement d'obstacle évolué, il est possible d'utiliser ces valeurs pour ralentir le robot en fonction de la proximité des obstacles. La figure 3.6 montre la couverture qui est fournie par ces capteurs sur le contour du robot. Les lignes droites illustrent la direction du champ de détection des différents capteurs. La flèche noire indique l'avant du robot. Une chose importante à considérer est que la zone couverte est très limitée

et il est possible pour le robot de percuter un obstacle sans même l'avoir détecté.

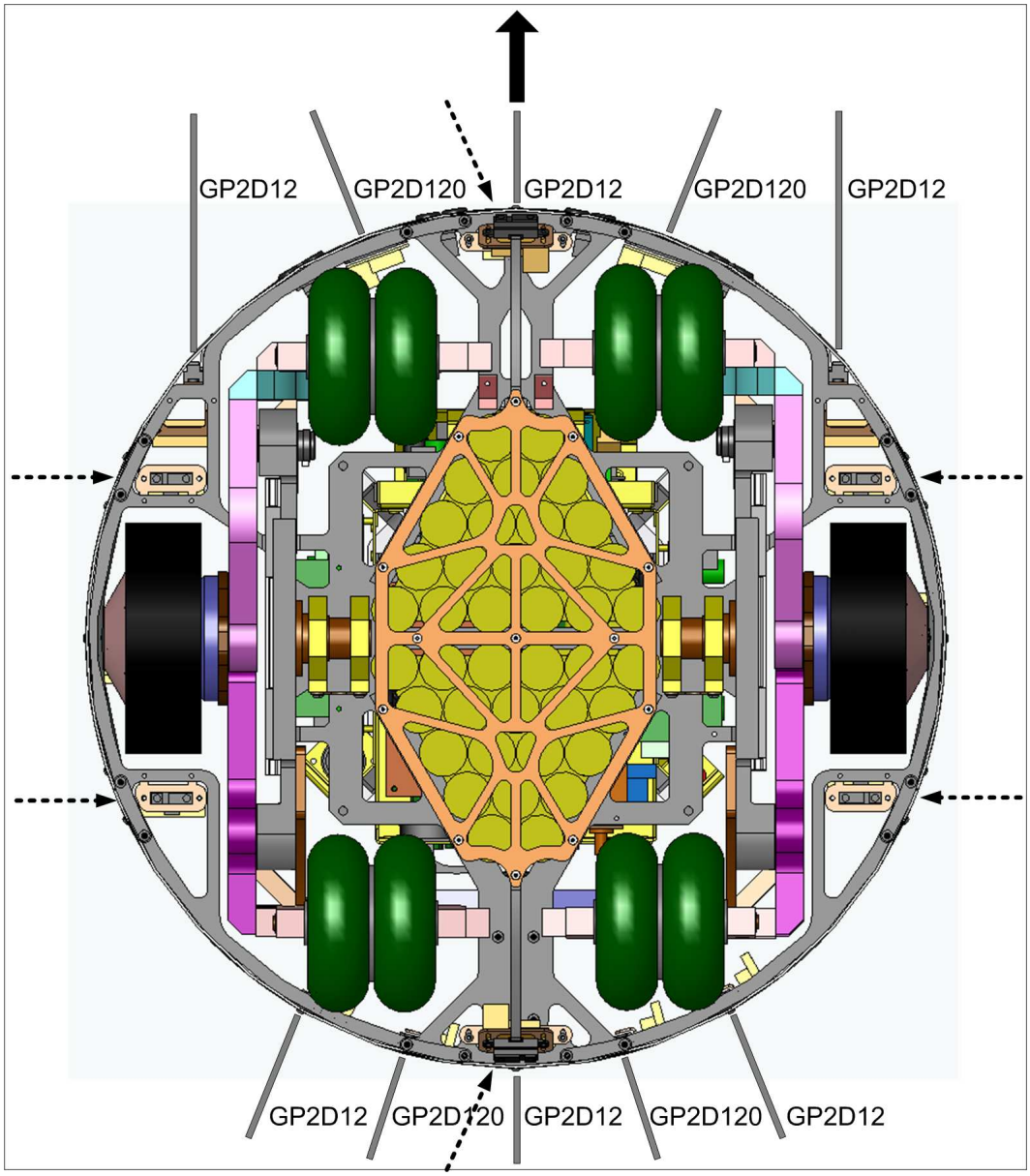


Figure 3.6 – Position des capteurs infrarouges sur le robot

La zone de détection des sonars, illustrée à la figure 3.7, montre que la zone de détection annoncée dans les spécifications de 30° se trouve à une distance d'environ 1 m. Le problème est qu'à courte distance (moins de 1 m), la zone de détection du sonar peut s'étendre jusqu'à 60°. Les zones de détection des sonars se chevauchent, ce qui fait que les lectures peuvent être influencées par le "crosstalk" si plusieurs sonars sont activés simultanément. La solution

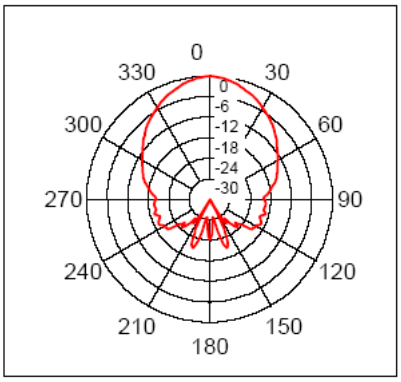


Figure 3.7 – Réponse du sonar SRF10

à ce problème est d'activer les sonars de manière séquentielle. À longue portée, la distance maximale de détection est de 11 m. En considérant que la vitesse du son est de 343 m/s, il faut un délai de 65 msec pour que l'écho revienne au capteur. Si l'on considère que le Télérobot a quatre sonars, on obtiendrait un ensemble de valeurs rafraîchies à chaque 260 ms, soit 3,85 Hz.

L'objectif est d'avoir un taux de rafraîchissement plus rapide (de l'ordre de 10 Hz). L'approche utilisée consiste à diminuer la distance maximale de détection des sonars. Dans le cas du projet, cette distance fut fixée à 1 m. La période alors requise pour une lecture est de 5,8 ms, et il est ainsi possible d'atteindre un rafraîchissement de 10 Hz. En considérant la valeur retournée par un sonar et la largeur de son cône de détection, il est impossible de connaître précisément la position de l'obstacle par rapport au robot. La figure 3.8 montre la couverture qui est disponible sur le Télérobot. La présence de zones de chevauchement (identifiées par les numéros 2, 4 et 6) sont des endroits où plusieurs senseurs peuvent détecter la présence d'un obstacle. À partir de ces zones, il est possible de déterminer une trajectoire qui permet au robot de ne pas se diriger directement vers l'obstacle. Les sonars rendent ainsi possible la mise en œuvre de ce type d'algorithmes car la couverture est quasi-complète devant le robot. Il est important de noter l'existence de zones "mortes" où les sonars ne peuvent détecter les obstacles. Elles se trouvent à courte portée entre les senseurs (zones blanches sur la figure 3.8).

Six capteurs infrarouges GP2D120 sont aussi utilisés pour détecter la présence de trous au

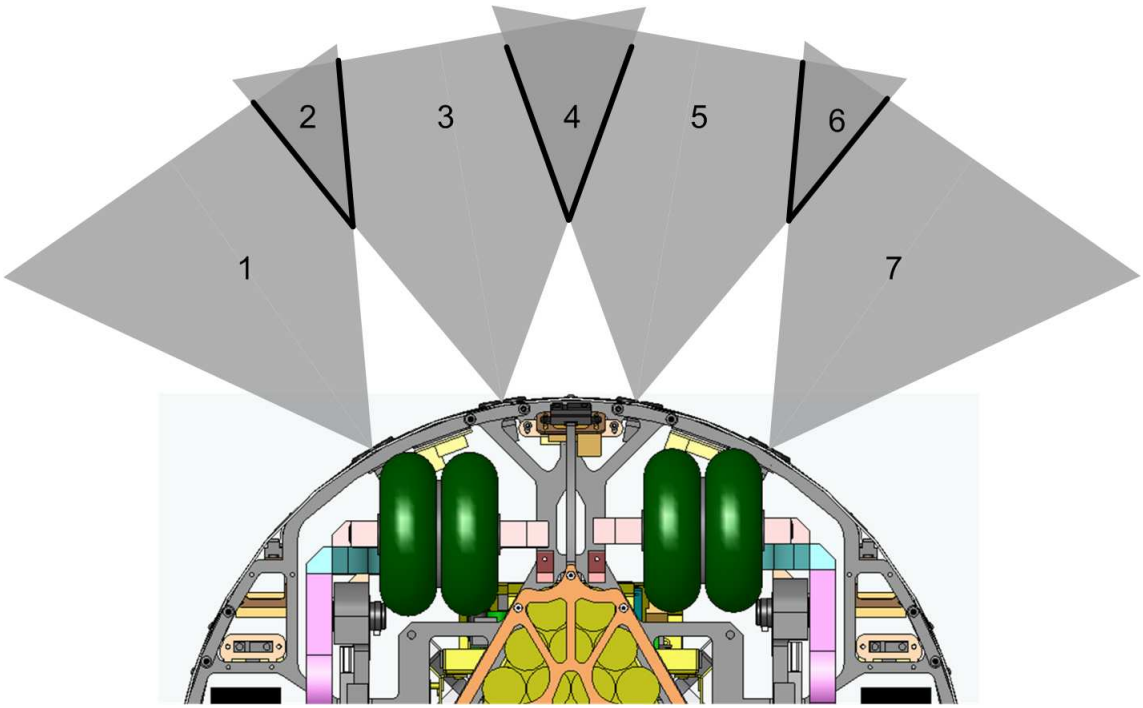


Figure 3.8 – Zones de détection offertes par les sonars

niveau du plancher. Il y en a un de chaque côté des roues de propulsion ainsi qu'un autre à l'avant et le dernier à l'arrière. La position des capteurs est identifiée par les flèches pointillées dans la figure 3.6. Il est ainsi possible au robot de percevoir une situation dangereuse comme quand le robot approche d'une cage d'escalier ou bien quand l'inclinaison du robot est trop importante. La couverture mise en place sur le Télérobot est différente de celle de Roomba. Nous avons choisis de mettre les capteurs près des roues de propulsion pour toujours être en mesure de pouvoir déplacer le robot en cas de problème. Les capteurs positionnés à l'avant et à l'arrière du robot reprennent la configuration du Roomba. Le couverture résultante est suffisante pour empêcher le robot de tomber dans un trou. Des tests, présentés à la section 5.3.2, en donnent la preuve.

Enfin, l'ajout d'encodeurs relatifs sur les bras de suspension nous a amené à ajouter une carte d'interface pour ces capteurs. Cette carte d'interface sert à lire les encodeurs relatifs placés sur les bras de suspension, permettant ainsi de connaître la position des roues et de mesurer la différence de hauteur entre les deux côtés du robot. L'objectif est de donner une information au niveau de la position des roues ainsi que sur la différence de hauteur entre

les deux cotés du robot. L'utilité de ces mesures et celles des infrarouges orientés vers le plancher, est d'identifier les situations où le robot pourrait basculer et tomber à la renverse. Cette situation n'est pas souhaitable car il n'y a aucun moyen automatique pour remettre le robot sur ses roues.

CHAPITRE 4

LOCALISATION, PLANIFICATION ET SUIVI DE TRAJECTOIRES

Télérobot doit être en mesure de se localiser dans l'environnement, de générer et de suivre une trajectoire. C'est probablement un des défis majeurs à surmonter dans l'usage du robot dans un domicile. Puisqu'à ce stade-ci du projet nos tests ne se font pas à l'intérieur de domiciles, l'objectif poursuivi pour nos travaux est de mettre en place les capacités perceptuelles et décisionnelles requises pour que cette fonctionnalité puisse être raffinée par la suite. Elle regroupe les étapes suivantes :

1. Localisation : le robot doit déterminer où il se trouve dans l'environnement.
2. Destination : le robot doit savoir où il doit se rendre. Dans notre cas, nous supposons que cette information lui est transmise par l'interface-utilisateur.
3. Représentation : le robot doit avoir une connaissance de l'environnement d'opération.
4. Navigation : le robot doit être en mesure de se déplacer par lui-même.

Les prochaines sous-sections viennent décrire les étapes de localisation, de représentation et de navigation telles que mises en œuvre sur Télérobot, avec leurs tests de performance associés.

4.1 Localisation

Cette section présente les résultats de tests de précision pour un système de localisation par odométrie et pour le système *NorthStar*. Les résultats obtenus sont comparés à ceux du système de localisation CARMEN [18], dont les principes sont expliqués à la section 4.1.1.

4.1.1 Localisateur CARMEN

Le logiciel CARMEN utilise une méthode probabiliste pour être en mesure de localiser le robot dans un environnement qui a été préalablement cartographié. Cette méthode est connue sous le nom de *Markov localization*. Le localisateur estime la probabilité que le robot se trouve à n'importe quelles position et orientation dans l'environnement. Au démarrage de l'application, il faut que l'utilisateur donne la position et l'orientation approximative du robot. À partir des informations retournées par le laser et des encodeurs de roues, l'application est en mesure de faire la mise à jour de la position du robot de manière autonome. Il aussi est possible à l'utilisateur de repositionner le robot en tout temps si le système est imprécis.

Dans le cas de notre implémentation, CARMEN utilise les valeurs retournées par le capteur laser de proximité SICK LMS200 et les valeurs fournies par les encodeurs de roues du TéléRobot. Pour la phase de tests, la carte utilisée couvre l'intérieur du LABORIUS ainsi que les corridors qui se trouvent à proximité. La représentation de l'environnement a été générée à partir des données du laser et de l'odométrie pendant que le robot se déplace dans la pièce.

Des tests de précision ont été faits dans le corridor devant le LABORIUS. Après un déplacement linéaire sur 5 m, l'estimation en X n'avait que 15 cm d'erreur et l'erreur d'orientation était de moins de 5°. Par contre, CARMEN a de la difficulté à positionner correctement le robot devant une surface uniforme comme un mur lisse. C'est l'estimation d'orientation qui est la plus affectée. Une situation similaire survient lorsqu'il y a une forme qui est impossible à identifier (e.g., présence d'une foule autour du robot). La méthode utilisée pour pallier au problème est d'ajouter des éléments fixes dans l'environnement pour augmenter la quantité de points de repères, et de limiter la présence d'individus dans l'aire d'expérimentation.

4.1.2 Odométrie

L'odométrie consiste à dériver la position $(X,\ Y)$ et l'orientation (Θ) à partir des lectures des encodeurs de roues. La lecture d'un encodeur de roue E s'exprime en nombre de pulses par secondes. Considérant qu'il y a P pulses (36000 sur Télérobot) par tour de roue et qu'une

roue de propulsion du robot a un rayon R (79.5 mm sur Télérobot), la vitesse d'une roue i (gauche ou droite) est donnée par la relation 4.1 :

$$V_i = E_i * (2\pi R/P) \tag{4.1}$$

La vitesse V résultante du robot s'évalue en effectuant une moyenne des vitesses des deux roues, selon la relation 4.2.

$$V = (V_{droite} + V_{gauche})/2 = (2\pi R/P) * (E_{droite} + E_{gauche})/2 \tag{4.2}$$

La vitesse angulaire du robot en radian/sec s'évalue à partir du différentiel de vitesse Θcycle entre les deux roues et la distance T (480 mm sur Télérobot) entre les deux roues de propulsion, selon la formule 4.3.

$$\Theta_{cycle} = (V_{droite} - V_{gauche})/T \tag{4.3}$$

Les variations de position (ΔX,ΔY) et d'orientation ($\Delta\Theta$) du robot se déterminent à partir de ces vitesses évaluées sur une intervalle de temps dt correspondant au cycle de rafraîchissement des données (10 Hz). Les équations 4.4, 4.5 et 4.6 sont utilisées pour corriger l'estimation de position. La présence de Θcycle est requise pour avoir une bonne précision lors de déplacement en arc. En ajoutant la moitié de Θcycle à l'orientation courante, nous venons d'ajouter une moyenne de déplacement lors du cycle courant. Autrement dit, les estimations des valeurs X et Y sont alors plus précises car le robot c'est déplacé tout en avançant.

$$X_{k+1} = X_k - (V_{cycle}dt * \sin(\Theta_k + \Theta_{cycle}/2)) \tag{4.4}$$

$$Y_{k+1} = Y_{k+1} + (V_{cycle}dt * \cos(\Theta_k + \Theta_{cycle}/2)) \tag{4.5}$$

$$\Theta_{k+1} = \Theta_k + \Theta_{cycle}dt \tag{4.6}$$

Pour fins de comparaison, des tests d'odométrie furent effectués sur un robot Pioneer 2,

fabriqué par la compagnie Mobile Robots Inc. et largement utilisé dans les laboratoires de recherche en robotique. Le premier test consiste à faire déplacer le robot en ligne droite sur différentes distances (500 mm, 1000 mm, etc.) et de comparer les déplacements réels du robot et la distance estimée par le robot. Des séries de dix tests furent réalisées pour ces distances, et la moyenne et ainsi que la variance des erreurs sont évaluées. Le tableau 4.1 présente les résultats obtenus pour le robot Pioneer 2 et le tableau **??** présente les résultats du Télérobot. En analysant les résultats à (1000 mm, 1500 mm et 2000 mm), on remarque l'accumulation de l'erreur en fonction de la longueur des déplacements des robots. Les erreur qui sont plus grandes pour la distance de 500 mm sont reliés à l'influence des glissements qui se produisent lors des départs et des arrêts du robot. Toutefois, l'erreur odométrique du Télérobot est de beaucoup inférieure à celle du Pioneer2. Les valeurs sont présentées en % car les distances réelles des différents tests n'étaient pas tout à fait les mêmes (+/- 15 mm). De cette façon, il est possible de comparer les différentes valeurs ensembles. Les résultats du Pioneer 2 peuvent sembler surprenants car l'erreur est supérieure pour une distance plus faible. La raison est que le robot a tendance à osciller lors des départs et des arrêts. Cette influence est alors diminuée sur des distances plus longues. Les distances de tests pour le Pioneer 2 ont été limitées à 1000 mm car à 1500 mm il était rendu difficile d'estimer précisement la position du robot car celui-ci avait tendance à tourner, ses roues de propulsion étant légèrement inclinées de par l'usure du système.

TABLEAU 4.1 – Exactitude de l'odométrie en ligne droite pour les robots sur des distances différentes

Robot	**Intervalle**	**Item**	**Essais**										**Moyenne**
Pioneer 2	500 mm	Erreur mm	8	6	6	8	1	2	1	19	7	7	6,5
		Intervalle réel mm	562	530	482	514	494	444	513	517	495	515	
		% d'erreur	1,423	1,132	1,245	1,556	0,202	0,450	0,194	3,675	1,414	1,359	1,265
Pioneer 2	1000 mm	Erreur mm	9	9	13	15	3	13	14	12	11	8	
		Intervalle réel	1016	988	994	1000	1055	1038	1030	1010	1022	1006	
		% d'erreur	0,885	0,911	1,308	1,500	0,284	1,252	1,359	1,188	1,076	0,795	1,056
Télérobot	500 mm	Erreur mm	4	2	0	6	1	1	2	5	2	1	2,4
		Intervalle réel mm	501	502	504	485	532	522	491	569	438	501	
		% d'erreur	0,798	0,398	0,000	1,237	0,188	0,192	0,407	0,879	0,457	0,200	0,476
Télérobot	1000 mm	Erreur mm	2	6	3	7	1	3	5	3	2	3	3,5
		Intervalle réel	1001	995	990	1052	1032	992	1006	980	1034	972	
		% d'erreur	0,200	0,603	0,303	0,665	0,097	0,302	0,497	0,306	0,193	0,309	0,348
Télérobot	1500 mm	Erreur mm	8	9	9	1	1	20	3	2	6	3	6,2
		Intervalle réel	1486	1496	1556	1532	1512	1478	1524	1492	1526	1514	
		% d'erreur	0,538	0,602	0,707	0,065	0,198	1,353	0,197	0,134	0,393	0,198	0,439
Télérobot	2000 mm	Erreur mm	12	9	6	7	6	7	3	10	13	8	8,1
		Intervalle réel	1990	2010	1985	2065	2028	1986	2013	1915	2044	1998	
		% d'erreur	0,603	0,448	0,302	0,339	0,296	0,352	0,149	0,577	0,636	0,400	0,410

Le second test sert à caractériser l'erreur sur l'orientation du robot à une vitesse de rotation de 12°/sec, et de mesurer la différence entre l'orientation initiale et l'orientation finale. Cette vitesse est la valeur minimale qui permet au robot Pioneer 2 de tourner de manière stable tout au long de sa révolution. Le tableau 4.2 présente les résultats observés, avec deux mesures obtenues par le robot Pioneer 2. La mesure P2OS est celle retournée directement par le microcontrôleur du robot Pioneer 2, tandis que la mesure corrigée est dérivée des vitesses des roues fournies par le P2OS (comme décrit par les relations 4.2 à 4.6 avec $T = 340$ mm pour le robot Pioneer 2). Les paramètres utilisés pour le Télérobot sont $T = 490$ mm et le rayon de la roue $R = 79{,}5$ mm. L'erreur odométrique de dix essais sur l'orientation du Télérobot est encore une fois bien inférieure à la meilleure des mesures obtenues avec le Pioneer 2.

TABLEAU 4.2 – Erreur sur l'estimation odométrique après une révolution (°)

Test	**P2(P2OS)**	**P2(corrigé)**	**Télérobot**
1	24,14	5,55	0,00
2	21,6	6,58	2,3
3	18,95	11	3,45
4	22,68	4,13	3,62
5	22,23	5,94	2,3
6	15,9	7,1	1,3
7	19,5	1,5	3,25
8	30,3	8,3	2,5
9	18,3	9,8	2,8
10	20	9	1,9
Moyenne (%)	5,933	1,914	0,651
Écart (°)	3,951	2,815	1,0898

Le troisième test consiste à exécuter une trajectoire précise et à mesurer les erreurs causées par les déplacements longitudinaux et latéraux du robot. Le robot se déplace sur un trajet en forme de triangle rectangle de 1,5 m par 2 m. Le tableau 4.4 présente la différence entre l'estimation de position avec la position réelle une fois que le robot retourne à son point de départ.

Une hypothèse expliquant les erreurs sur l'odométrie Télérobot est la variation du paramètre T en fonction de la surface d'opération et la largeur de la surface de contact des roues.

Les roues de propulsion du robot ont une largeur de 50 mm, ce qui crée une plus grande surface de contact avec le sol, mais aussi une plus grande sensibilité aux défauts de surface de l'environnement. L'ajout d'élastiques de 15 mm de largeur sur les roues de propulsion permet de réduire la surface de contact avec le sol, et du même coup, les erreurs. Le tableau 4.4 présente les erreurs observées avec l'ajout d'élastiques sur les roues. Ces mesures confirment une amélioration en Y et en angle, mais pas en X. Ceci est normal car la réduction de la surface de contact peut occasionner une moins bonne friction dans les déplacements selon l'axe longitudinal de la roue. L'usage de roues profilées pour réduire davantage la surface de contact avec le sol pourrait améliorer l'odométrie du robot. En guise de comparaison, des mesures ont aussi été prises avec le localisateur CARMEN utilisant le laser comme capteur de proximité. Évidemment, ces erreurs sont beaucoup plus petites que celles de l'odométrie, mettant en évidence le gain de ne pas seulement se fier à l'odométrie pour la localisation. Noter que l'odométrie est fusionnée avec les données laser dans CARMEN pour la localisation. Les erreurs moyennes fournies par CARMEN nous permettent de localiser de manière adéquate le robot dans l'environnement. Il est également possible de constater que les erreurs d'odométrie sont très variables selon le cas. Les mesures présentées au tableau 4.4 confirment une amélioration en Y et en angle, mais pas en X. Ceci est normal car la réduction de la surface de contact peut occasionner une moins bonne friction dans les déplacements selon l'axe longitudinal de la roue.

TABLEAU 4.3 – Erreurs à 250 mm/sec et 28°/sec (sans élastique)

Test	Erreur odométrique			Erreur de CARMEN		
	X (mm)	**Y (mm)**	°	**X (mm)**	**Y (mm)**	°
1	52	13	2,095	6	12	1,095
2	78,5	1	4,088	6,5	11	1,088
3	15,5	74	2,192	23,5	22	0,192
4	141	35	2,200	5	9	0,800
5	49	5,5	3,647	6	12,5	0,647
6	79,5	12	3,877	15,5	5	0,877
7	76	21	3,989	5	11	0,989
8	83	7,5	1,855	12	15,5	0,145
9	93	25	2,647	13	15	0,647
10	112,5	28,5	2,640	16,5	24,5	0,640
Moyenne	78	22,25	2,923	10,9	13,75	0,712

TABLEAU 4.4 – Erreurs à 250 mm/sec et 28°/sec (avec élastique)

Test	Erreur odométrique			Erreur de CARMEN		
	X (mm)	**Y (mm)**	°	**X (mm)**	**Y (mm)**	°
1	57,5	34	3,851	13,5	8	0,149
2	52	27	4,566	17	14	0,566
3	54,5	53,5	5,675	3,5	8,5	0,325
4	33,5	25	6,239	8,5	7	0,239
5	47	44,5	5,395	29	19,5	1,395
6	37,5	42	4,457	12,5	17	0,457
7	53,5	21,5	4,239	20,5	19,5	2,239
8	6	13	1,655	13	12	0,655
9	26,5	63,5	7,803	24,5	21,5	0,803
10	35,5	30	4,021	11,5	6	0,021
Moyenne	40,35	35,4	4,790	15,35	13,3	0,685

Le dernier test réalisé consiste à évaluer les erreurs de la localisation à des vitesses plus basses. Les résultats, présentés au tableau 4.5, montrent des erreurs plus grandes avec les vitesses plus faibles. La précision des mesures des encodeurs de roues à basse vitesse en est la cause.

4.1.3 Système de balises infrarouges *NorthStar*

Ce système est composé d'un projecteur placé dans une zone à couvrir et projetant des signaux infrarouges au plafond. On place sur le robot un détecteur capable d'identifier les signaux infrarouges et le dispositif est en mesure de se repérer dans la zone à partir de ces signaux. Comme le montre la figure 4.1, les signaux infrarouges doivent se trouver dans le champ de perception du détecteur. L'étendue de la zone est fonction de la distance H entre le détecteur et le plafond : plus H est grand, plus la zone couverte W est grande. Le manufacturier recommande que le ratio W/H soit inférieur ou égal à 1,4 pour avoir les performances optimales. Un ratio de 2,4 donne la distance de détection maximale. Les spécifications techniques sont données au tableau 4.6. La distance recommandée est celle qui répond au ratio de 1,4 et la distance maximale répond au ratio de 2,4.

De plus, une inclinaison du détecteur peut occasionner des variations de 150 mm et de 2° par degré d'inclinaison. Des tests sommaires dans les domiciles ont permis de confirmer ces

TABLEAU 4.5 – Erreurs à 100 mm/sec et 12°/sec (avec élastique)

Test	Erreur odométrique			Erreur de CARMEN		
	X (mm)	**Y (mm)**	°	**X (mm)**	**Y (mm)**	°
1	55,5	35	4,543	12,5	11	0,543
2	64	133	7,132	5	10	0,868
3	53,5	97	7,977	6,5	7	0,977
4	47,5	82	4,891	8,5	7	0,109
5	57,5	45	3,977	11,5	9	0,977
6	54,5	55	4,109	18,5	15	0,109
7	44	56	3,783	12	9	0,217
8	59,5	57	6,543	7,5	12	0,543
9	71	72	5,217	7	7	0,217
10	68,5	64	5,977	10,5	8	0,977
Moyenne	57,55	69,6	5,415	9,95	9,5	0,553

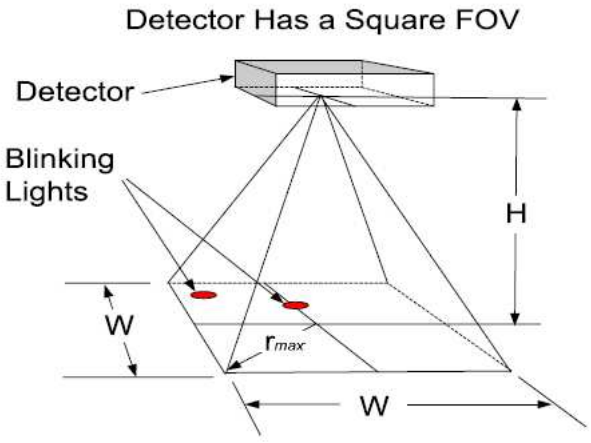


Figure 4.1 – Zone couverte par le *NorthStar* (le graphique est inversé verticalement)

TABLEAU 4.6 – Spécifications techniques du système *NorthStar*

Métrique	**Distance recommandée**	**Distance maximale**
Position(cm)	1 - 4	10 - 40
Angle(°)	1 - 2	2 - 8

valeurs. Le système demande une période de 100 msec afin de calculer la position du robot (X, Y, Θ) à l'intérieur de la zone de détection. Jusqu'à 10 zones distinctes peuvent être définies en plaçant plusieurs projecteurs pour couvrir une plus grande région.

L'interface de communication entre le Télérobot et le système *NorthStar* est fait via un port série RS-232 à 1200 baud. En considérant les délais de transmission des messages et le temps de lecture du capteur, le temps de réponse minimal est de 400 msec.

De toute évidence, les performances du système sont affectées par le type de surface (telles que confirmées par la compagnie Acroname[1]) et la présence d'objet au plafond. Par exemple, le laboratoire est doté d'un plafond suspendu avec des lumières à tubes fluorescents et des bouches d'aération qui provoquent des différences au niveau de la hauteur du plafond. Le tableau 4.7 montre les performances observées en laboratoire avec une valeur de $H = 1800$ mm, ce qui revient à dire que la distance recommandée est inférieure à 1,26 m de chaque côté du point de référence. La distance maximale est de 2,26 m. La zone de test se trouve sous une partie de plafond suspendu qui ne comporte pas de luminaire ou de bouches d'aération directement au-dessus du récepteur. Les valeurs présentées sont la différence donnée entre la lecture du *NorthStar* et de la distance réelle qui a été mesurée manuellement entre la position du récepteur et le point de référence (0,0) de la zone de répérage. Des mesures ont été prises à chaque 100 mm sur une distance de 2,0 m pour un total de vingt points.

TABLEAU 4.7 – Intervalles des erreurs du *NorthStar* en laboratoire

Métrique	**Distance recommandée**	**Distance maximale**
Position (cm)	1 - 9	1 - 16
Angle (°)	1 - 9	1 - 7

Une série de tests a aussi été effectuée en trois points de l'environnement afin de comparer

[1]http ://www.acroname.com/robotics/info/evaluation/northstar/northstar.html

les performances entre ce système et CARMEN. Il est important de mentionner que l'un des points utilisés pour l'analyse est celui où le système *NorthStar* a été calibré, autrement dit l'endroit où les résultats sont optimaux. De plus, ce point se trouve sous une partie du plafond qui se trouve au même niveau que le point de référence. Le tableau 4.8 montre une comparaison sur dix lectures entre les deux systèmes. Il est possible de voir que les deux systèmes ont une performance comparable dans ces conditions d'opération.

TABLEAU 4.8 – Comparatif du *NorthStar* et de CARMEN au point de référence

Mesure	Estimation du *NorthStar*			Estimation de CARMEN		
	X (mm)	**Y (mm)**	°	**X (mm)**	**Y (mm)**	°
1	8956	11140	0	9000	11150	359,993
2	8956	11140	0	8980	11140	0,175
3	8953	11136	0	8980	11140	0,329
4	8956	11116	0	8980	11150	0,783
5	8955	11113	0	8980	11150	0,462
6	8956	11114	0	8970	11150	0,923
7	8956	11112	0	8960	11150	1,13305
8	8956	11110	0	8970	11150	0,636
9	8956	11113	0	8960	11150	1,028
10	8957	11120	0	8960	11150	0,818
Moyenne	8956	11121	0	8974	11148	0,698

Un autre test a été effectué en un point où le niveau du plafond est différent du point de référence. Cette situation n'est pas présente dans la plupart des domiciles sauf lorsque le robot passe sous un cadre de porte. Le tableau 4.9 montre l'erreur causée par le niveau du plafond. L'erreur propore du *NorthStar* n'est alors plus négligeable.

4.2 Planification de trajectoire

La planification de trajectoire s'effectue à partir d'une représentation de l'environnement. Pour les fins du projet, l'accent n'est pas mis sur les capacités du planificateur de trajectoire mais bien sur son intégration sur le Télérobot. Nous avons choisi d'utiliser une approche par décomposition en cellules exactes [6]. Elle consiste à diviser l'espace libre de l'environnement en cellules rectangulaires, comme le montre la figure 4.2 pour l'espace d'expérimentation au

TABLEAU 4.9 – Comparatif du *NorthStar* et de CARMEN en un point statique

Mesure	Estimation du *NorthStar*			Estimation de CARMEN		
	X (mm)	**Y (mm)**	°	**X (mm)**	**Y (mm)**	°
1	7430	11530	171,161	7028	11304	180
2	7430	11550	171,978	7029	11303	180
3	7420	11550	172,447	7009	11304	180
4	7430	11550	171,768	7029	11304	180
5	7420	11550	172,174	7029	11304	180
6	7420	11550	172,3	7020	11303	180
7	7430	11530	171,915	7003	11303	180
8	7430	11530	172,118	7025	11304	180
9	7420	11550	172,09	6998	11305	180
10	7420	11540	172,398	7006	11303	180
Moyenne	7425	11543	172,035	7018	11304	180

LABORIUS. La division présentée est celle qui a été utilisée lors des tests avec l'interface usager du Télérobot. Elle définit l'environnement du LABORIUS ainsi que les locaux voisins. Chaque cellule est définie par les coordonnées X, Y dans le plan de ses quatres sommets, desquels il est possible de déterminer un point central. Le rôle du planificateur de trajectoire est de déterminer un trajet passant par les points centraux des cellules, partant de l'endroit où se trouve le robot vers la destination désirée.

Avec ces cellules, il est possible de représenter les liens entre elles à partir d'un graphe d'adjacence, comme le montre la figure 4.3. Ce graphe définit un espace de recherche pour l'établissement de trajectoires. Pour déterminer un trajet, il suffit de spécifier une destination qui sera associée à la cellule où elle se trouve. Un algorithme de recherche vient alors déterminer la trajectoire à réaliser pour minimiser la distance à parcourir. Nous avons choisi d'utiliser un algorithme A* [7] avec trois heuristiques d'évaluation :

– *Greedy* : utilise la distance entre les centres de la cellule de départ et de la cellule de destination, même si ces cellules ne sont pas adjacentes ;
– *Dijkstra* : utilise la distance entre les centres des cellules sur l'ensemble de la trajectoire ;
– *Mixte* : recherche qui regroupe les deux critères précédents.

Un test a été fait pour voir l'impact des différentes heuristiques sur le temps de traitement de l'algorithme. Le tableau 4.10 montre les résultats pour une recherche entre la cellule C4

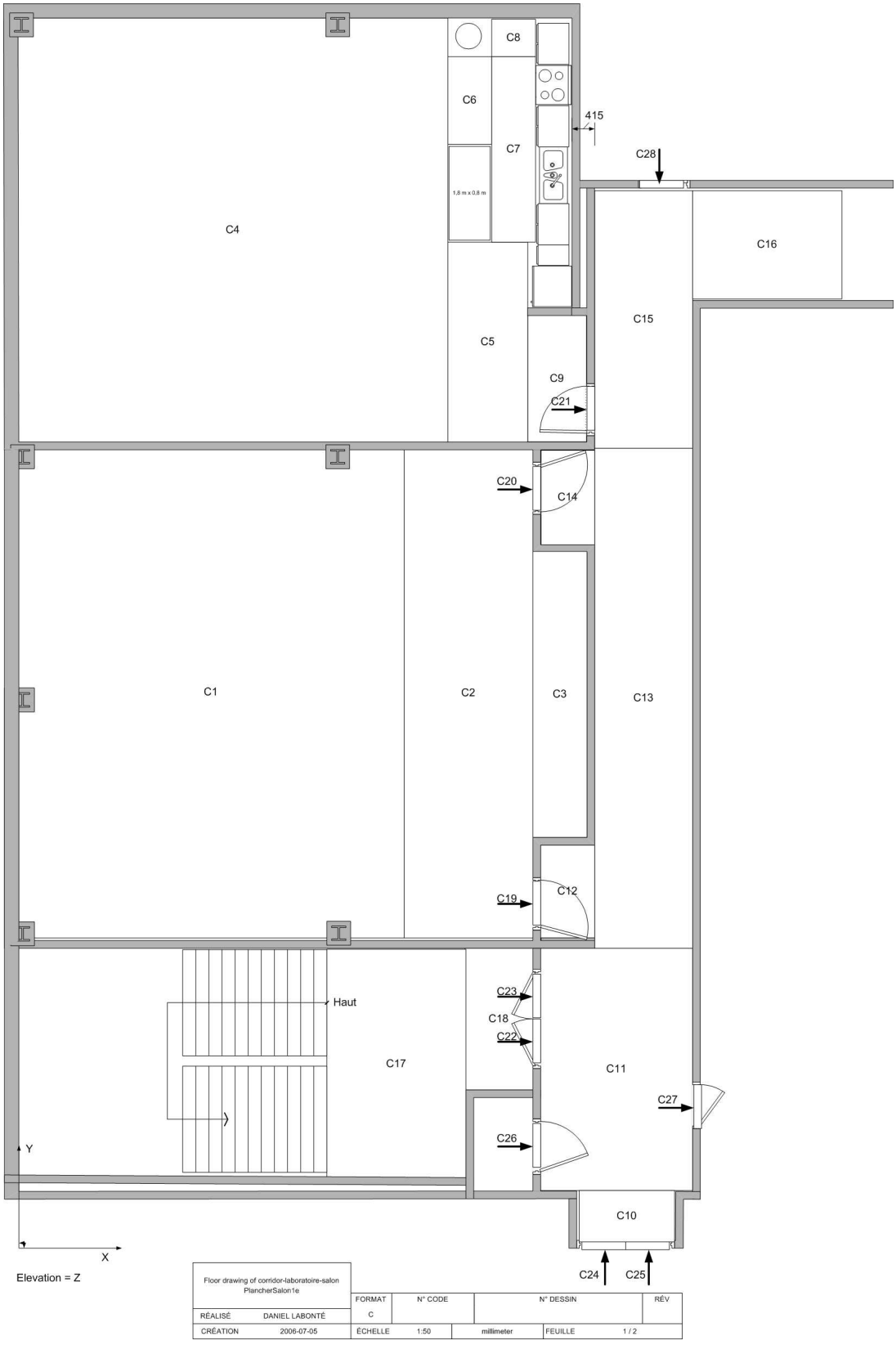


Figure 4.2 – Division en cellule de l'environnement d'expérimentation

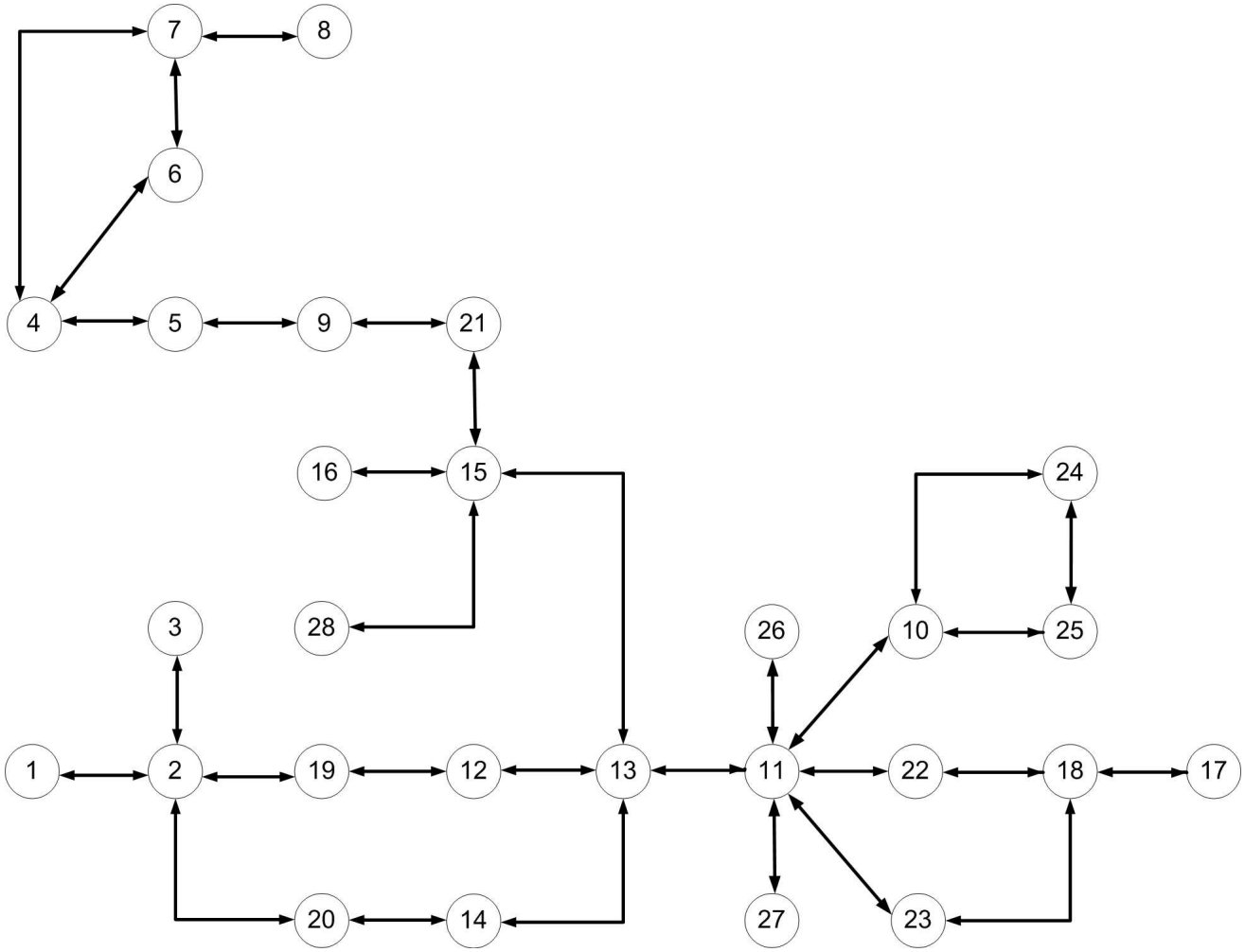


Figure 4.3 – Arbre d'adjacence pour l'environnement d'expérimentation

et la cellule C13. La présence de variations est reliée au fait que le système ne fonctionne pas de manière synchrone. Il y a des processus qui sont plus prioritaires et qui viennent prendre du temps de traitement au besoin. Il est quand même possible de voir qu'en moyenne, les performances de la méthode mixte sont supérieures aux deux autres. La différence entre les différentes méthodes serait accentuée avec un environnement plus complexe (plus de cellules). Dans l'ensemble des cas, le trajet résultant est identique.

Notons que les tests ont été effectués sur le robot Pioneer 2, qui a un ordinateur embarqué plus lent que sur le Télérobot. La raison est que les différences entre les méthodes sont alors accentuées dans cette configuration.

TABLEAU 4.10 – Temps de recherche de trajets selon le mode en msec

Essai	**Mixte**	***Greedy***	***Dijkstra***
1	4,27	18,07	5,11
2	3,25	3,24	3,11
3	3,16	10,67	10,58
4	10,3	3,12	3,11
5	3,55	9,51	3,06
6	3,19	3,09	3,05
7	3,17	3,11	11,27
8	3,17	3,19	3,10
9	3,25	3,16	3,05
10	4,86	7,13	3,14
Moyenne	4,34	6,11	4,85

4.3 Discussion

Pour ce qui est de la localisation, il est possible de voir qu'il n'y a pas de système parfait. L'odométrie ne peut pas être utilisée seule car la fiabilité de ce module décroît avec le temps. Le système *NorthStar* ne peut pas être utilisé seul non plus car l'influence de l'environnement (e.g., surface du plafond) rend l'estimation de position inutilisable. Le système qui donne les meilleures performances est CARMEN. Un inconvénient est que le capteur laser utilisé présentement avec CARMEN (SICK LMS200) est coûteux à l'achat et en consommation énergétique. Une alternative à étudier est l'utilisation du capteur laser URG04 avec

CARMEN. Ce capteur a une faible consommation (2,5 W) et un coût à l'achat d'environ 1500$ US. Les nouvelle versions du logiciel CARMEN sont également à envisager dans le but d'améliorer les performances actuelles (à partir de 0.6-beta disponible depuis juin 2006). D'autres alternatives comme l'utilisation du vSLAM devront être analysées car ceci pourrait être utilisé pour complémenter CARMEN. La fusion de différentes sources de localisation pourra alors être envisagée afin d'avoir un système flexible et fonctionnel dans différents types d'environnement.

CHAPITRE 5

CONCEPTION LOGICIELLE POUR LA PRISE DE DÉCISION

Télérobot se veut être une plate-forme robotique de téléopération. Indépendamment de son interface-utilisateur, la plate-forme doit pouvoir recevoir des commandes d'un ordinateur distant et lui retourner des informations (e.g., flux vidéo et audio, données des capteurs). Ces échanges s'effectuent par communication TCP d'Internet, suivant un protocole propre au Télérobot et présenté à l'annexe A. Cependant, Télérobot doit aussi faire preuve d'autonomie décisionnelle pour jouer de façon appropriée son rôle dans le domicile. L'ensemble des fonctionnalités identifiées pour Télérobot sont présentées dans la liste suivante :

1. Acheminer l'état des capteurs et de sa prise de décision à l'interface-utilisateur ;
2. Interpréter et appliquer les commandes (déplacement, modes, arrêt, pousser des objets) envoyées par l'interface-utilisateur ;
3. Utiliser un mode sécuritaire de déplacement par évitement d'obstacles ;
4. Se localiser dans l'environnement, de générer et d'y suivre une trajectoire ;
5. Identifier le besoin de recharge des batteries et aller se recharger ;
6. Détecter la présence d'un bris de communication avec l'interface-utilisateur afin d'arrêter le robot.

La conception logicielle pour la mise en œuvre de ces fonctionnalités comporte l'élaboration d'une architecture décisionnelle et d'une architecture logicielle. L'architecture décisionnelle décrit les modules de traitement et leurs interrelations pour la prise de décision. L'architecture logicielle explique comment ces modules sont implémentés sur le robot.

5.1 Architecture décisionnelle

L'architecture décisionnelle utilisée sur le Télérobot répond aux caractéristiques des architectures hiérarchiques hybrides [23] [1]. La figure 5.1 illustre le principe général de ces architectures. Le Planificateur de trajet, expliqué à la section 4.2, se trouve au sommet de la hiérarchie. Il sert à générer des trajectoires en fonction des objectifs du robot. Le module Exécutif permet de faire le suivi de l'exécution d'une mission du robot, comme la rencontre des différents points d'un trajet à suivre. Le module de contrôle temps réel est composé des Comportements, activés et configurés par le module Exécutif et de contrôleurs PID pour les commandes aux actionneurs du robot. Sur Télérobot, le Planificateur de trajet, l'Exécutif et les Comportements sont implémentés sur l'ordinateur Pentium M 1,7 GHz, tandis que les contrôleurs sont implémentés à même les microcontrôleurs PIC18F décrit au chapitre 3. Les comportements acheminent à une fréquence de 10 Hz les consignes aux contrôleurs.

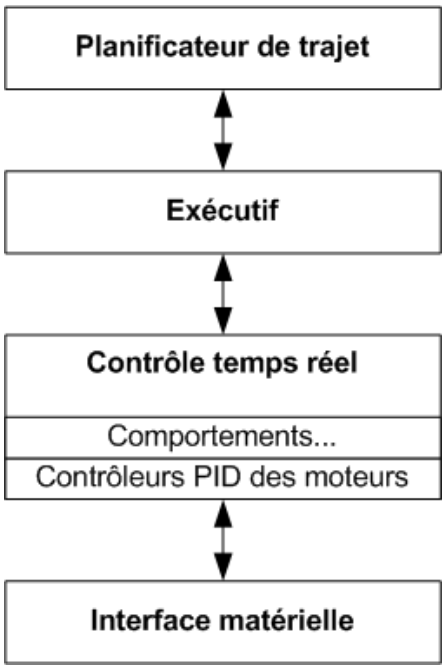


Figure 5.1 – Exemple d'architecture appliqué à un problème de navigation autonome

La figure 5.2 illustre plus spécifiquement l'architecture hiérarchique hybride mise en œuvre pour Télérobot. Le bloc Localisation sert à localiser le robot à partir des capteurs d'odométrie, du *NorthStar* ou du laser. Cette information est requise pour la planification et le suivi

(par le Planificateur de trajet) et l'exécution (par le comportement *Goto*) de trajectoires. Le module Exécutif vient activer les comportements en fonction des commandes provenant de l'interface-utilisateur (GUI) et de la perception du robot. Six comportements servent à contrôler les actions (vitesse de déplacement linéaire en mm/sec et vitesse de rotation en 1/100 °/sec du robot), via un bloc d'arbitration qui les priorises.

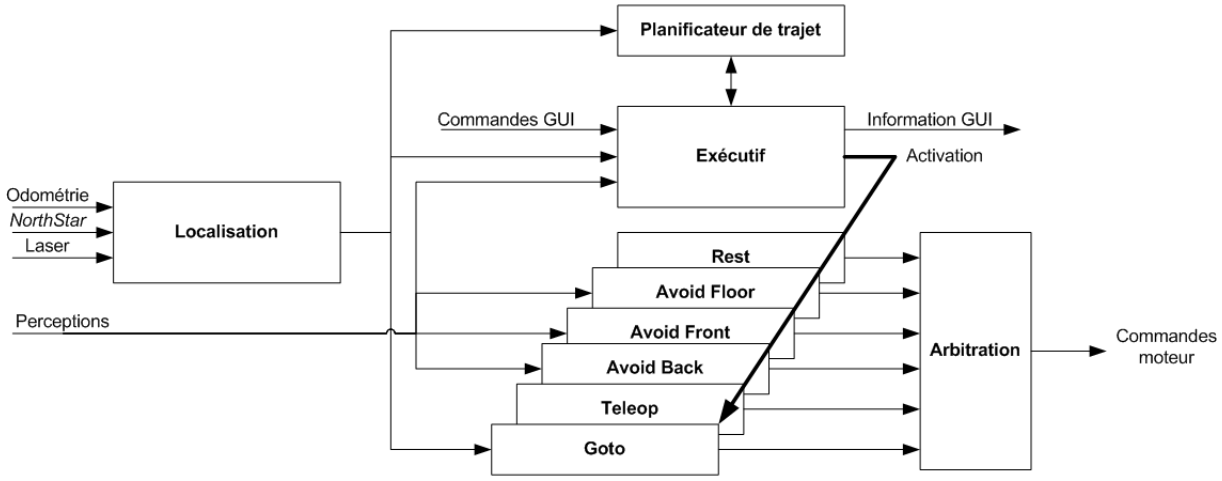


Figure 5.2 – Architecture décisionnelle utilisée sur Télérobot

5.1.1 Module Comportements

La figure 5.3 illustre le schéma-bloc du module Comportements, qui inclut les six comportements et leur arbitration par priorité (le comportement le plus haut étant plus prioritaire que les comportements du dessous). Chaque comportement peut envoyer des consignes (Vitesse longitudinale, Vitesse de rotation), ou ne rien envoyer si aucune commande n'est à générer. Les comportements sont :

- *Rest* : Envoi par défaut une consigne (0,0) lorsqu'il est activé.
- *Avoid Floor* : Détecte la présence du plancher. Freine le robot lorsqu'un trou à l'avant ou à l'arrière du robot est détecté. L'objectif est d'éviter que le robot tombe dans un trou ou dans une cage d'escalier.
- *Avoid Front* : Détecte la présence d'un obstacle à l'avant du robot. Le robot diminue sa vitesse longitudinale lorsque l'obstacle le plus près est à une distance de 50 cm. Il arrête complètement quand l'obstacle est à 20 cm et moins. Un algorithme linéaire est utilisé

pour ralentir le robot entre ces deux valeurs limites. Ce sont les mesures retournées par le capteur laser qui sont utilisées par cet algorithme, ce qui donne 120 échantillons espacés d'un degré. Le cône résultant couvre +/- 60 ° par rapport à l'avant du robot. Le choix de cette configuration est basé sur la précision des mesures qui sont utilisés par l'application CARMEN pour fonctionner de façon adéquate. La figure **??** montre l'étendue de la zone couverte par le capteur laser. FIGURE LASER ICI

- *Avoid Back* : Détecte la présence d'un obstacle à l'arrière du robot. L'idée d'une zone où le robot ralentit est reprise sauf que la limite extérieure est fixée à 30 cm. Le comportement utilise les valeurs des capteurs infrarouges positionnés à l'arrière du robot (3 GP2D12 et 2 GP2D120). Ceux-ci sont illustrés à la figure **??**. Le ralentissement du robot est calculé de la même façon que pour le comportement *Avoid Front*.
- *Teleop* : Envoi comme consigne les valeurs de vitesse reçues via les commandes de déplacement. Le comportement permet également de limiter les vitesses demandées en fonction des vitesses maximales spécifiées par le bloc Exécutif. Il y a deux ensembles de vitesse maximales permises (linéaire, rotation). Le premier ensemble est en mode de fonctionnement normal (250 mm/sec , 28°/sec) et l'autre en mode "PUSH" (100 mm/sec , 12°/sec). Dans le deuxième cas, les vitesses permises sont plus faibles car le robot se déplace sans avoir de comportement d'évitement d'obstacle.
- *Goto* : Calcule la trajectoire à suivre pour permettre au robot d'atteindre un point précis de l'environnement en fonction de la position actuelle du robot dans l'environnement. La première étape est d'orienter le robot en direction de l'objectif. Ensuite, le robot se déplace linéairement jusqu'à une certaine distance du point à atteindre, avec une tolérance de 200 mm. Ceci correspond à environ 35% par rapport à la superficie du Télérobot, et est considérée comme acceptable pour l'application. La dernière étape consiste à orienter le robot selon la valeur spécifiée dans la définition du point à atteindre. Cette étape est optionnelle et elle est configurable à même le comportement ou en spécifiant un angle égal ou supérieur à 360°. Les vitesses de déplacement maximales sont les mêmes que pour le comportement Teleop.

Notons que, pour l'instant, les comportements *Avoid Front* et *Avoid Back* ne font qu'arrêter le robot. Ce choix fut fait pour laisser un plus grand contrôle à l'opérateur du robot lors des

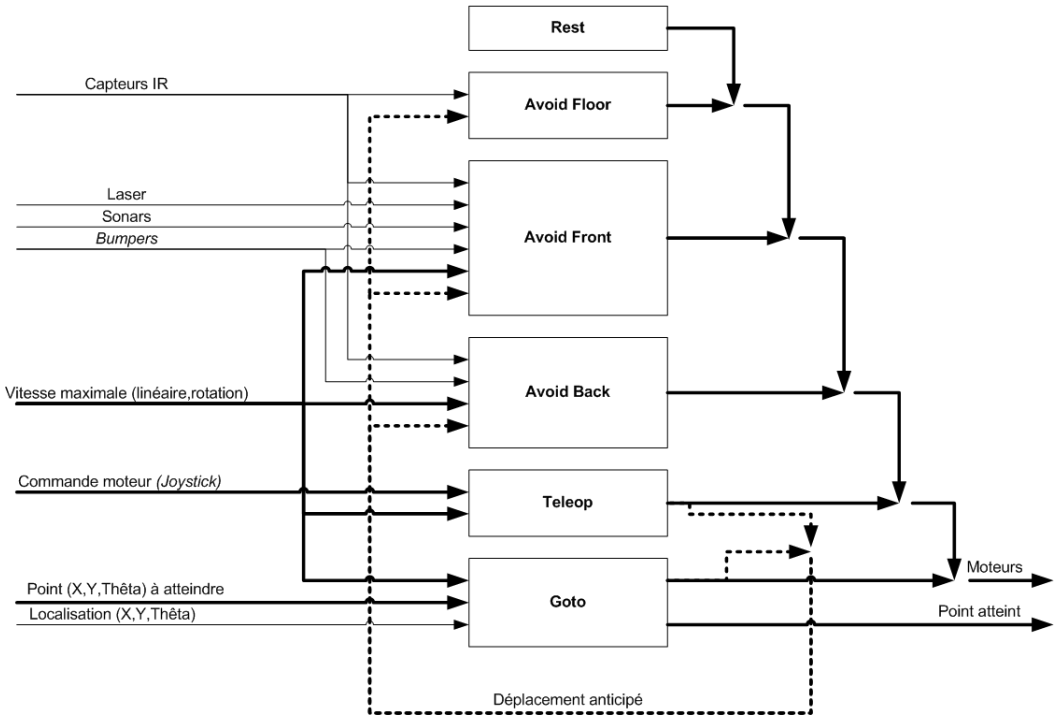


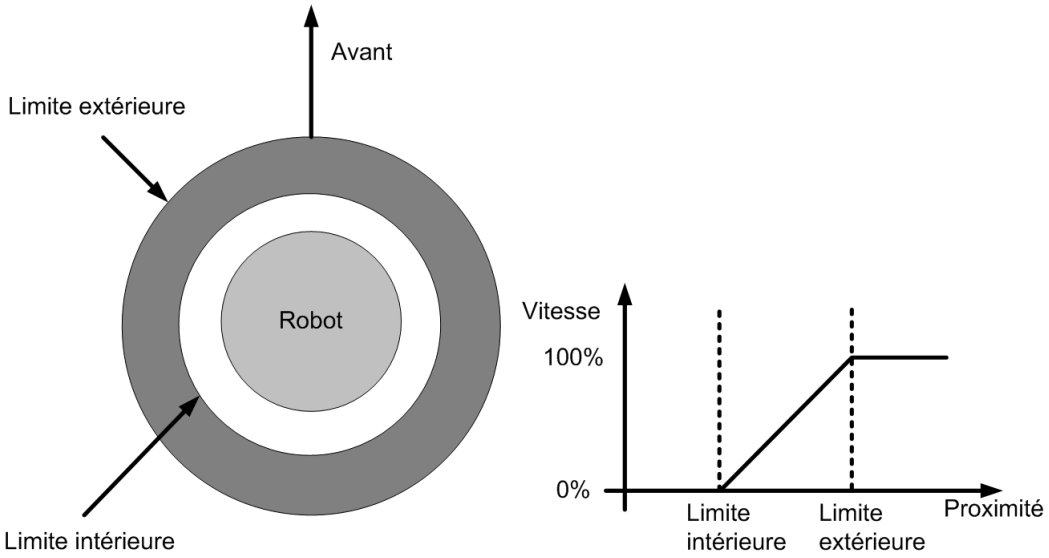
Figure 5.3 – Implémentation du module Comportements

tests à réaliser sur l'interface-utilisateur. L'approche utilisée est présentée à la figure 5.4. La vitesse maximale est permise quand l'obstacle se trouve au-delà de la distance extérieure. La vitesse maximale permise décroît de manière linéaire entre les deux distances limites et elle est nulle quand l'obstacle est à une distance plus faible que la limite intérieure.

Des comportements plus sophistiqués pourront éventuellement être élaborés en utilisant les informations des comportements *Goto* et *Teleop* pour influencer la direction à prendre pour éviter les obstacles.

5.1.2 Module Exécutif

La figure 5.5 illustre le schéma-bloc du module Exécutif. Le module *Navigate* gère la réalisation de trajets à partir des points soumis par le Planificateur de trajet. Il fournit le premier point du trajet (X, Y, Θ) au comportement *Goto*. Celui-ci gère le déplacement du robot pour atteindre l'objectif et envoie une confirmation au bloc Exécutif lorsque le point est considéré comme étant atteint. Dans ce cas, le module *Navigate* envoi le prochain point du trajet et



(a) Zone de proximité (b) Vitesse maximale permise

Figure 5.4 – Vitesse maximale permise en fonction de la proximité des obstacles

le cycle recommence jusqu'à ce que le trajet soit complété. Les modules *Survive* et *Lonely* assurent les conditions d'opération sécuritaires du robot en activant le comportement *Rest*. Le module *Survive* examine le niveau de tension des batteries du robot et il active *Rest* quand le niveau de tension est inférieur à 23,5 V pendant 10 secondes (pour éviter d'être sensible aux variations de tension causées par l'activation des moteurs). Ce choix se justifie par la contrainte de vouloir rester sécuritaire avec le robot, le temps que les modules de navigation autonome soient validés en conditions réelles. En temps normal, ce module viendrait activer un comportement de recharge autonome et la génération d'une trajectoire vers une station dont le robot connaît la position. Le module *Lonely* identifie une faute lorsqu'aucune communication n'est reçue avec l'interface-utilisateur pendant 300 msec, activant alors *Rest* pour des fins de sécurité.

Le module *BehaviorActivation* regroupe les règles d'activation des comportements en fonction des informations obtenues de l'interface-utilisateur (*Commandes GUI*) et des autres modules de l'Exécutif. Une commande de déplacement longitudinal ou latéral provenant de l'interface-utilisateur se traduit par l'activation du comportement *Teleop* avec des consignes maximales de vitesse qui dépendent du mode d'opération courant (normal ou "PUSH") . Le comportement *Goto* est activé lorsqu'il y a au moins un point d'un trajet à atteindre

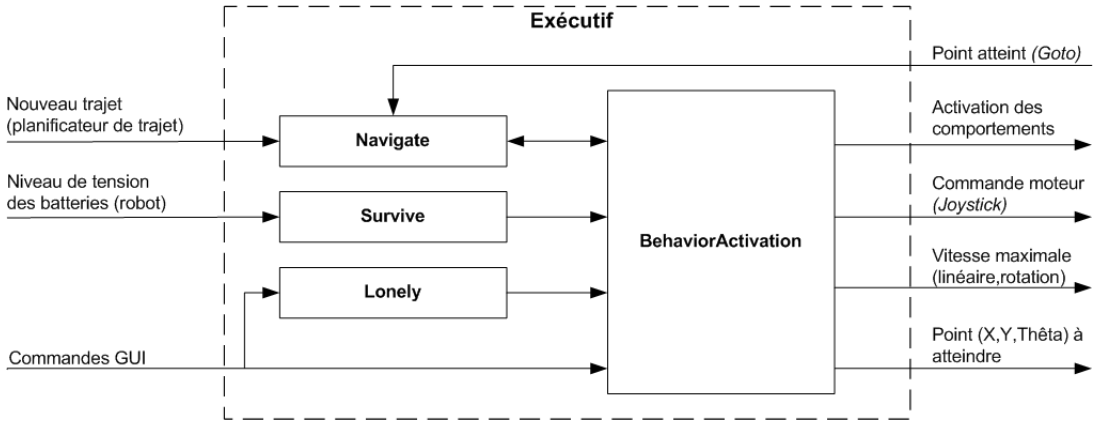


Figure 5.5 – Schéma-bloc du module exécutif

et que le robot est localisé de façon absolue dans l'environnement. Une commande de suspension via l'interface-utilisateur du mode de suivi de trajectoire autonome par le robot influence l'exécution de la trajectoire (*Goto*). Une requête d'arrêt d'urgence active le comportement d'arrêt (*Rest*). Les comportements *Avoid Floor*, *Avoid Front* et *Avoid Back* sont activés par défaut. L'activation du mode *Push* sur l'interface-utilisateur désactive le comportement d'évitement d'obstacles avant (*Avoid Front*) tout en limitant les vitesses maximales de déplacement permises.

5.2 Architecture logicielle

La mise en œuvre de l'architecture décisionnelle est effectuée avec FlowDesigner [9] et MARIE [8]. FlowDesigner est un environnement de programmation graphique avec des outils de visualisation et de déverminage facilitant la programmation robotique. MARIE facilite la réutilisation de modules logiciels entre les applications robotiques. Cet outil offre des modalités de communication et d'interfaces standardisées entre des applications logicielles. Des composants logiciels servent pour le transfert des données entre les modules logiciels. Les *Splitter* permettent d'envoyer une donnée sur différents ports de communication. Les *Mailbox* permettent de mémoriser une valeur d'entrée et les autres composants MARIE peuvent récupérer la donnée par une requête à la *Mailbox*. Chaque lien de communication utilise des numéros de port TCP différents pour permettre d'isoler les transferts d'information.

La figure 5.6 illustre la mise en œuvre de l'architecture décisionnelle sous MARIE. Les numéros de ports TCP utilisés sont identifiés à même les liens, avec *a* pour signifier le port accepteur de connection et *c* pour le connecteur. L'initialisation des modules se fait en activant d'abord le côté accepteur et ensuite le connecteur. Les blocs de format (..*fd*..) identifie des blocs programmés sous FlowDesigner. Les blocs utilisés sont :

– TRGUIManagerAA : ce bloc sert de lien bidirectionnel de communication entre l'interface-utilisateur et le robot. Il est régi selon le protocole décrit à l'annexe A. L'échange d'information se fait sous format texte afin de faciliter le déverminage et les ajustements au protocole. Les commandes de l'utilisateur sont interprétées seulement si le message reçu est complet et valide. Le critère est qu'il faut que le numéro d'identification du message soit le même au début et à la fin de la séquence de commande.
– aafdTRDecisionNetwork : ce bloc renferme le bloc Exécutif ainsi que les comportements présentés à la section 5.1.1. Les interactions entre les différents modules ainsi que la priorisation des sorties des comportements sont gérées dans ce bloc.
– aafdLocalize : ce bloc localise le robot dans son environnement en fonction des informations provenant de différents capteurs. Différentes méthodes sont possibles :
    1. Odométrie uniquement (voir section 4.1.2)
    2. *NorthStar* uniquement (voir section 4.1.3)
    3. CARMEN (voir section 4.1.1). Notons que le point de référence de la carte (Xo,Yo,Θo) de la carte avec CARMEN est positionné à une valeur constante et différente de (0,0,0°), position qui correspond à l'initialisation du point de référence pour l'odométrie (i.e., le point de départ du robot correspond au point (0,0,0°)). Il faut donc ajouter les points de référence Xo,Yo de la carte avec CARMEN à celles de l'odométrie pour positionner le robot par rapport avec cette carte. Une compensation par rapport à Θo n'est pas utilisée car la position de départ du robot pour les tests ainsi que celle utilisée pour la génération de la carte était la même. Ce qui fait que l'orientation était contrôlée en tout temps, ce qui enlève le besoin de compenser ce paramètre.
    4. Combinaison de méthodes. Il serait possible de combiner l'odométrie à *NorthStar* et à CARMEN en implémentant les algorithmes appropriés dans ce module.
– aafdPathPlanner : c'est dans ce bloc que la planification de trajet s'effectue, en prenant

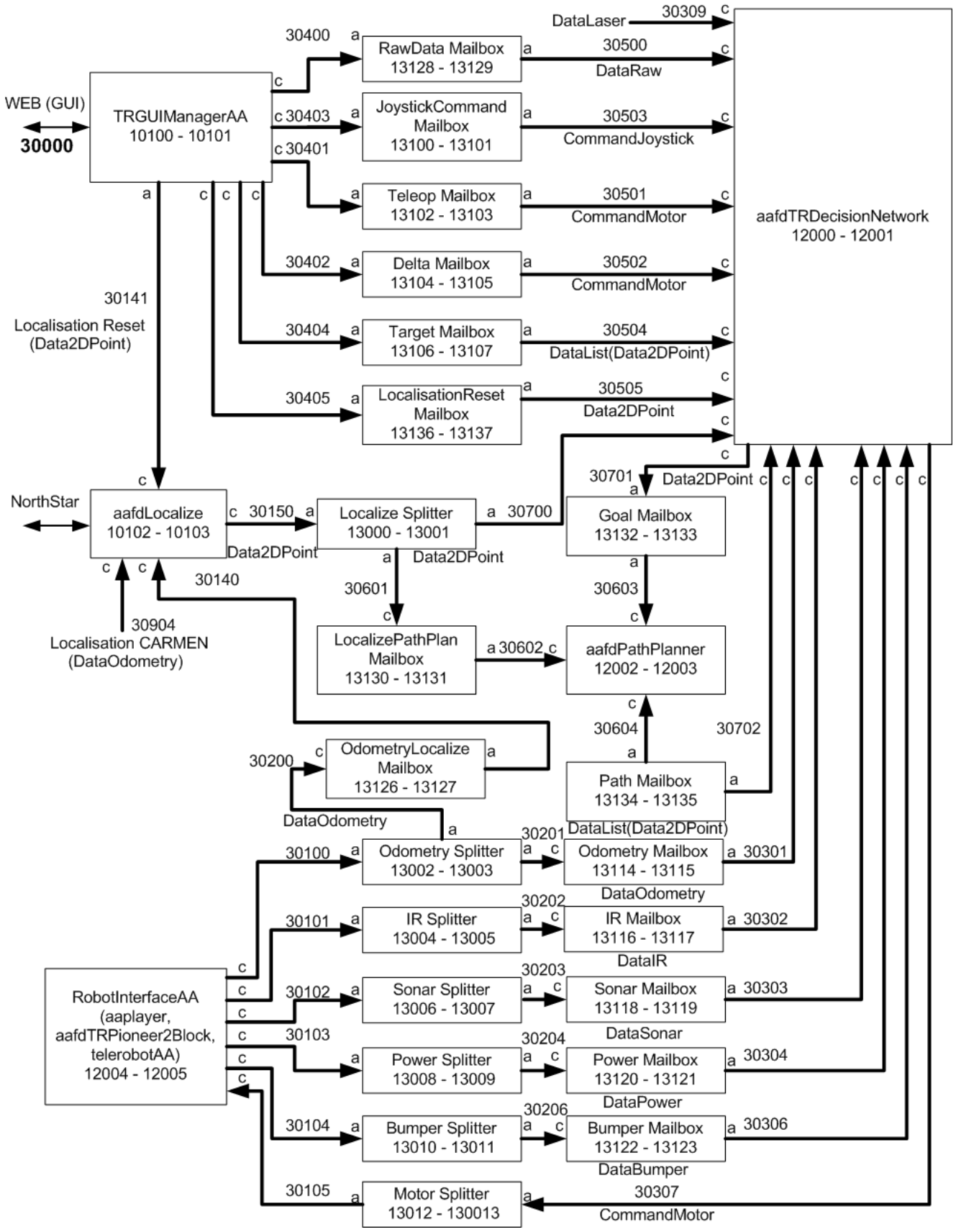


Figure 5.6 – Implémentation sous MARIE de l'architecture décisionnelle de Télérobot

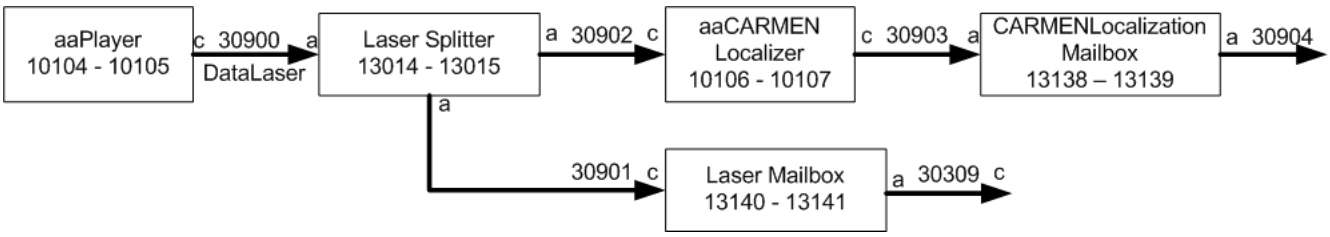


Figure 5.7 – Implémentation du laser et CARMEN

compte de la position estimée du robot et de la position de l'objectif (*Goal*). Il est important que les deux valeurs soient référencées à un même système de coordonnées. Ce bloc retourne une liste de points (X, Y, Θ) à franchir.

– RobotInterfaceAA : le mécanisme de ce bloc est déterminé en fonction de l'utilisation du système en simulation ou avec le robot réel. En effet, pour faciliter les développements, un lien avec le logiciel Stage [26] est possible via MARIE. Pour le robot réel, les commandes sont acheminées aux microcontrôleurs PIC18F via le bus CAN du robot.

Afin d'établir la correspondance entre l'architecture décisionnelle et la figure 5.6, le tableau 5.1 présente la correspondance entre les modules décisionnels et les blocs de traitement. Enfin, le flux d'information *LocalisationCARMEN* est dérivé de la chaîne de traitement présentée à la figure 5.7. Dans cette chaîne, le programme *Player* [26] offre une interface qui permet de communiquer avec le laser. Les 180 valeurs (une par degré) sont envoyées au *aaCARMENLocalizer*, le localisateur de CARMEN.

TABLEAU 5.1 – Correspondance entre l'architecture décisionnelle et l'architecture logicielle

Modules décisionnels (figure 5.2)	**Blocs de traitement logiciel (figure 5.6)**
Localisation	aafdLocalize
Planificateur de trajet	TRPathPlanner
Perceptions	RobotInterface
Commandes moteur	RobotInterface
Exécutif	aafdTRDecisionNetwork
Comportements	aafdTRDecisionNetwork
Arbitration	aafdTRDecisionNetwork

5.3 Performances du système

Des tests ont été faits pour valider le bon fonctionnement de l'architecture et pour en caractériser les performances. Pour l'ensemble des tests présentés, l'architecture complète était active sur le robot. Autrement dit, les valeurs sont prises en considérant des conditions d'opération du système intégré. Les tests se divisent en trois sections. La première cherche à établir le temps de réaction de l'architecture décisionnelle suite à la réception de différentes commandes de téléopération. La deuxième contient les preuves de fonctionnement des différents comportements moteurs. La troisième démontre le fonctionnement global de l'algorithme de suivi de trajet autonome.

5.3.1 Temps de réaction de l'architecture logicielle

Les temps de réaction du système ont été analysés pour permettre de caractériser les performances de l'architecture. Le premier test consiste à mesurer sur dix essais le temps entre la réception d'une commande d'arrêt d'urgence logiciel et le moment où la commande moteur, reçue par le bloc RobotInterfaceAA, passe à une vitesse nulle. Le tableau 5.2 montre les résultats des dix essais, présentant une moyenne de 158 msec.

TABLEAU 5.2 – Temps de réaction suite à la réception d'une commande d'arrêt d'urgence (valeurs en msec)

	Essais									
Délai	214	151	136	137	106	190	174	164	111	192

Le deuxième test consiste à évaluer le temps nécessaire entre le moment où la commande de vitesse change, au niveau du TRGUIManagerAA, et le moment où la nouvelle consigne de vitesse est reçue au bloc RobotInterfaceAA. Le tableau 5.3 montre le temps de réaction de l'architecture suite à un changement sur la commande de vitesse (passage de 0 mm/sec à 250 mm/sec) sur dix essais. La moyenne observée du temps de réaction est de 132 msec.

Les temps de réaction moyen ne dépassent donc pas 200 msec, soit deux cycles de traitement. Les variations sont dues au fait que l'ensemble des blocs logiciels fonctionnent de façon

TABLEAU 5.3 – Temps de réaction pour un changement de vitesse (valeurs en msec)

	Essais									
Délai	164	67	110	174	192	82	175	145	85	123

asynchrones.

Le dernier test de ce bloc consiste à caractériser le temps entre la réception d'une commande de suivi d'un trajet autonome (GPP) et le moment où le robot débute son déplacement. Le délai moyen observé est de 656 msec. Le tableau 5.4 illustre également le temps requis par l'algorithme de recherche de trajet (*TRPathPlanner*) pour générer le trajet à suivre. Il est à noter que le temps de recherche pour l'algorithme de recherche de trajets (1.447 msec) est négligeable dans le résultat. Ces tests ont été effectués en situation optimale de fonctionnement, c'est à dire que l'algorithme n'a pas de recherche à faire et trouve la solution immédiatement car les deux points sont dans la même cellule. Ce choix fut fait pour permettre de bien voir le temps qui est nécessaire pour la communication entre les blocs de l'architecture. Des tests ont été faits sur Télérobot avec l'élaboration d'un trajet qui couvre différentes cellules. Le temps additionnel pour la recherche et l'élaboration d'un trajet est d'environ 1 msec ce qui est négligeable devant le délai complet de l'architecture qui est en moyenne 656 msec. Le délai complet de l'architecture ne dépasse donc pas une seconde, ce qui est un temps de réaction correct pour ce type de commande.

TABLEAU 5.4 – Temps de réaction suite à une demande de trajet autonome

Délai (msec)	**Temps recherche (msec)**
573	2,08
548	1,378
820	1,378
766	1,346
729	1,363
622	1,376
692	1,356
326	1,387
814	1,425
675	1,377

5.3.2 Preuves de fonctionnement des comportements moteurs

Dans le cas du comportement d'arrêt d'urgence (*Rest*), des tests ont été effectués pour vérifier le délai entre le temps où la commande d'arrêt d'urgence est reçue et le moment où les encodeurs de roues confirment que le robot est complètement arrêté. Les tableaux 5.5 et **??** montre les résultats de dix essais en fonction de deux vitesses initiales distinctes. Les moyennes observées sont de 371 msec pour une vitesse initiale de 100 mm/sec et de 515 msec à une vitesse initiale de de 250 mm/sec. Ceci correspond à des distances maximales théoriques d'arrêt de 37,1 mm et 128,75 mm respectivement. En considérant le pire cas (737 msec), la distance maximale théorique franchie est de 184 mm. Il est à noter que les cartes de contrôle des moteurs n'utilisent pas de pente dans l'asservissement des vitesses. Ce mode de fonctionnement est utilisé par le robot lors de situations normales de fonctionnement. La coupure est faite de façon brusque, c'est pour cette raison que les temps de réaction sont aussi rapides.

TABLEAU 5.5 – Temps de réaction pour l'arrêt complet à partir de deux vitesses initiales (valeurs en msec)

Délai (vitesse initiale 100mm/sec)	**Délai (vitesse initiale 250mm/sec)**
326	536
281	737
290	592
291	484
439	434
485	338
337	341
490	598
287	544
484	541

Pour les tests de validation des comportements d'évitement d'obstacles, la méthode utilisée est de déplacer le robot en ligne droite pour qu'il rencontre un obstacle qui se trouve directement devant lui. La distance mesurée est celle entre le châssis du robot et l'obstacle. Dans les trois cas (Avoid Floor, Avoid Front, Avoid Back), la technique utilisée pour arrêter le robot est de définir une zone où le robot ralentit en fonction de la distance de l'obstacle, telle qu'illustrés à la figure 5.4.

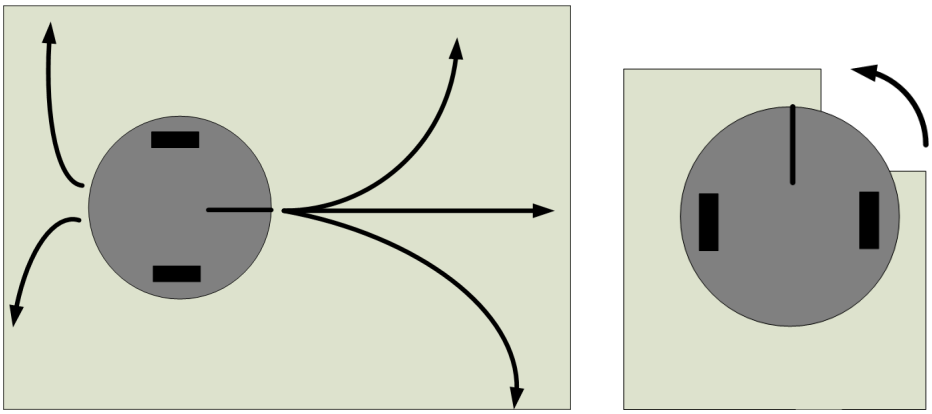
Dans le cas du comportement *Avoid Front*, la limite extérieure est placée à 500 mm et la limite intérieure à 100 mm. Ces valeurs sont calculées par rapport au châssis du robot. Ces valeurs ont été sélectionnées suite à des tests d'opération dans l'environnement de test. Elles sont donc qualitatives et elles devront être changées en fonction de l'environnement d'opération. Les tests ont été effectués à trois vitesses différentes pour voir les limites du système. Le tableau 5.6 montre les valeurs moyennes obtenues suite à dix essais pour chacune des vitesses. Un fait à noter dans ce tableau est que les distances d'arrêt sont supérieures à 100 mm. Ceci est relié à l'algorithme utilisé pour le contrôle de vitesse en fonction de la distance des obstacles. La consigne de vitesse qui est envoyée aux contrôleurs des moteurs est trop faible pour que les moteurs tournent.

TABLEAU 5.6 – Comparaison des distances d'arrêt pour le comportement *Avoid Front*

Vitesse linéaire (mm/sec)	**Distance moyenne de l'obstacle (mm)**
100	140
250	110
350	110

Dans le cas du comportement *Avoid Back*, les capteurs de proximité infrarouges, orientés vers l'arrière, sont utilisés pour la détection d'obstacles. Une série de dix essais a été effectuée à une vitesse de -100 mm/sec et la distance moyenne d'arrêt est de 100 mm. Les paramètres utilisés pour la zone de ralentissement étaient de 200 mm et 100 mm. La spécification est donc atteinte. Il est à noter que si la vitesse maximale permise est augmentée, il faut augmenter la limite extérieure définissant la zone de proximité (voir figure 5.6).

Dans le cas du comportement *Avoid Floor*, c'est également les capteurs infrarouges qui sont utilisés pour mesurer la distance entre le robot et le sol. Les tests de fonctionnement ont été faits en déplaçant le robot sur une planche de bois qui a une épaisseur de 12,7 mm qui est placée sur le sol. La figure 5.8 montre les deux types de déplacements qui ont été testés. Le premier test consite à déplacer le robot dans toute les directions et de vérifier que le robot ne descends pas de la planche. Le deuxième est de vérifier que les roues de propulsion ne tombent pas dans un trou suite à une rotation. Dans les deux cas, les tests ont été concluants suite des déplacements qui ont été faits par commandes de téléopération directes (*joystick*). Une



(a) Déplacements en arc (b) Déplacements en rotation

Figure 5.8 – Déplacements de la plate-forme sur la planche

analyse de distance a été faite en considérant des déplacements en ligne droite vers le bout de la planche (mouvements avant et arrière). La vitesse initiale du robot est de 250 mm/sec soit la vitesse maximale permise pour le Télérobot. La distance moyenne d'arrêt, sur dix essais, entre le centre des roues de propulsion et le bord de la planche est de 300 mm. La distance de référence utilisée est de 200 mm pour considérer que le robot se déplace sur une surface sécuritaire. Le comportement bloque la vitesse à 0 lorsque la distance devient trop grande (c'est à dire en présence d'un trou).

Enfin, le test de fonctionnement pour le comportement *Goto* consiste à mesurer la distance à laquelle le robot arrête par rapport à la position du point à atteindre. Le tableau 5.7 donne la position estimée par CARMEN où le robot s'arrête en fonction du but à atteindre. Différentes vitesses de déplacement et de surfaces de roulement sur les roues du Télérobot ont été testées. Les distances données sont une moyenne sur dix essais. Il est possible de confirmer que le comportement *Goto* fonctionne selon les spécifications.

TABLEAU 5.7 – Distance entre le point d'arrêt du robot et l'objectif

Vitesse linéaire	**Vitesse de rotation**	**Présence d'élastiques**	**Distance (mm)**
250	28	non	188
100	12	non	194
250	28	oui	188

Le dernier test consiste à analyser la trajectoire qui est suivie par le robot par rapport

au trajet optimal dans un environnement d'opération réel. Le trajet consiste à atteindre trois points différents sur des distances de 2 m, 1,5 m et 2,5 m. La figure 5.9 montre la position du robot (X,Y) par rapport au trajet optimal entre différents points. Il est important de mentionner que les trois points sont compris dans une même cellule, le trajet optimal entre les différents points est tout simplement un déplacement en ligne droite. Les vitesses maximales utilisées sont de 250 mm/sec et 28°/sec. Le localisateur CARMEN est utilisé comme source de postionnement par le robot. Il est possible d'observer que le robot ne se rend pas exactement aux différents sommets du triangle. Ceci est dû à la tolérance qui est utilisée par le comportement pour considérer le point comme étant atteint qui est de 200 mm. La distance moyenne entre la position réelle du robot sur le sol et celle de l'objectif est de 189 mm après dix essais. Ce qui confirme que le comportement fonctionne de façon adéquate.

5.3.3 Test de validation du suivi de trajet autonome

La dernière série de tests sert à valider le fonctionnement du système de navigation autonome de façon globale. La capacité du Télérobot à suivre un trajet dans un environnement réel a été démontré. Il faut maintenant faire la preuve que le robot est en mesure de générer et de suivre un trajet qui passe par différentes cellules. Des tests ont été faits en utilisant l'environnement de simulation *Stage*. La raison pour utiliser cet environnement est qu'il est difficile de faire passer le Télérobot dans des endroits étroits (e.g., cadre de porte). Ceci est dû au comportement d'évitement d'obstacle (*Avoid Front*) qui bloque le robot quand il se trouve trop près d'un obstacle. Une autre raison est qu'il n'y a pas de différences au niveau du code utilisé pour la génération et le suivi de trajet. La seule différence dans l'architecture est le bloc *RobotInterfaceAA* qui active les moteurs du Télérobot ou dans l'autre cas donne les commandes de déplacements au robot simulé.

L'environnement de test proposé est illustré à la figure 5.10. Cet environnement est simple et il est divisé en quatre cellules distinctes. L'arbre d'adjacence des cellules est également montré dans cette figure. La définition qui a été faite de l'environnement est montrée dans le tableau 5.8. Les vitesses maximales qui sont utilisées par le robot lors de ses déplacements sont de 250 mm/sec et 28°/sec et la tolérance utilisée pour considérer l'objectif visé comme

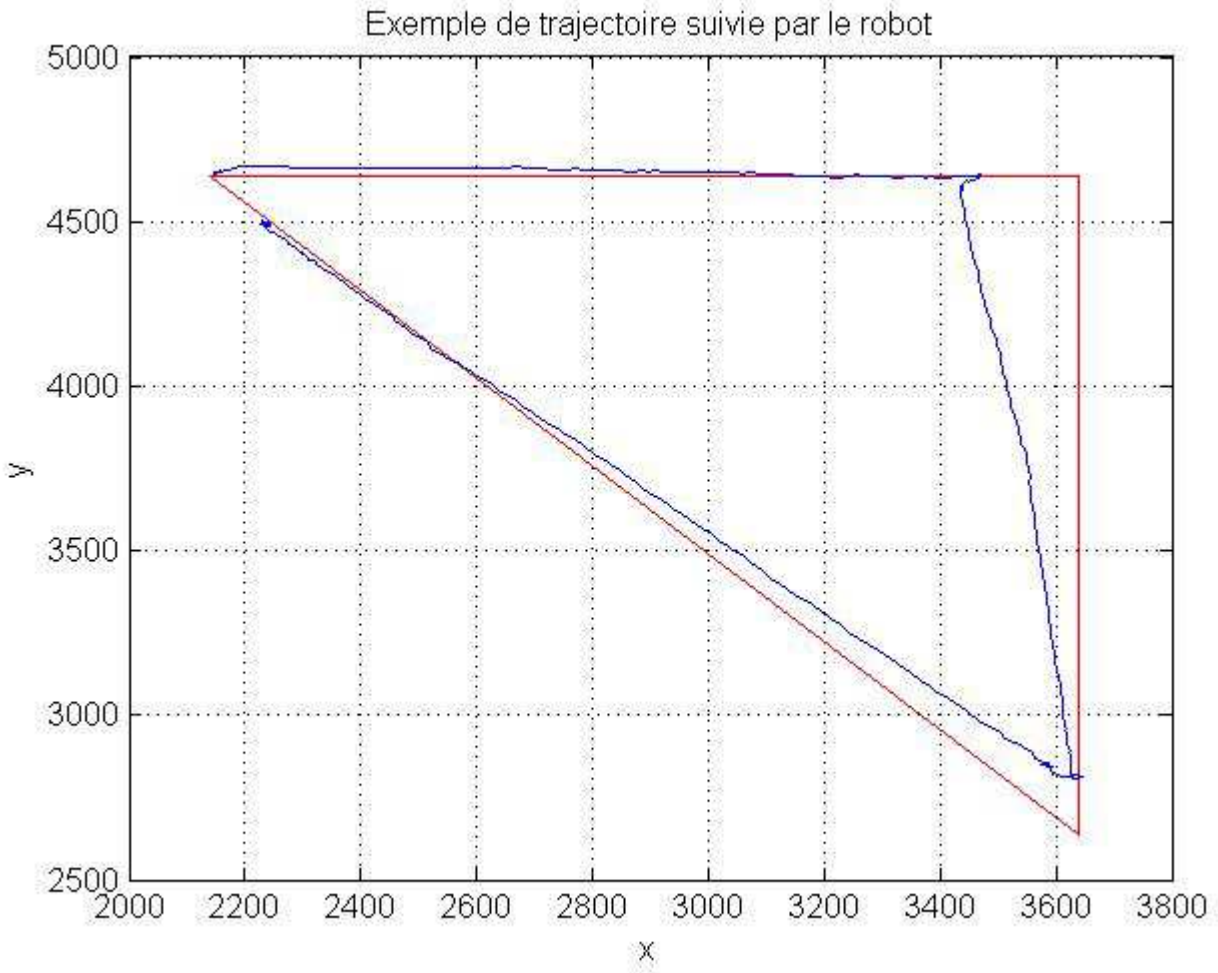
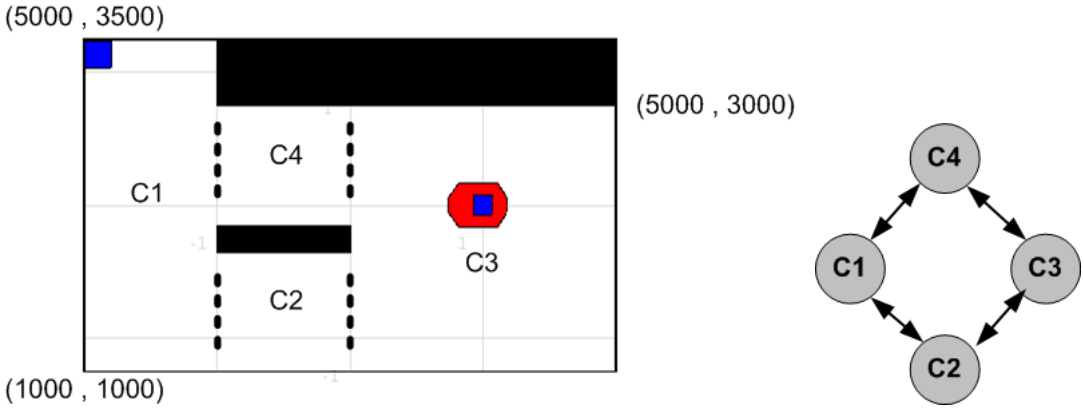


Figure 5.9 – Position du Télérobot par rapport à la trajectoire optimale

étant atteint est de 100 mm.

La figure 5.11 montre le résultat de la génération et du suivi de deux trajectoires pour un même objectif qui se trouve à la position (1500 , 2500). La différence entre les deux examples est le point de départ du robot (4000 , 2250) et (4000 , 1500) respectivement. Le critère de sélection pour établir trajet est de minimiser la distance linéaire qui sera couverte par le robot. Le calcul de distance pour le premier segment est fait en fonction de la distance entre le centre des cellules adjacentes et la position actuelle du robot et non la position du centre de la cellule où le robot se trouve. C'est pourquoi l'algorithme prends les chemins les plus directs pour se rendre à l'objectif.



(a) Division de l'environnement (b) Arbre d'adjacence

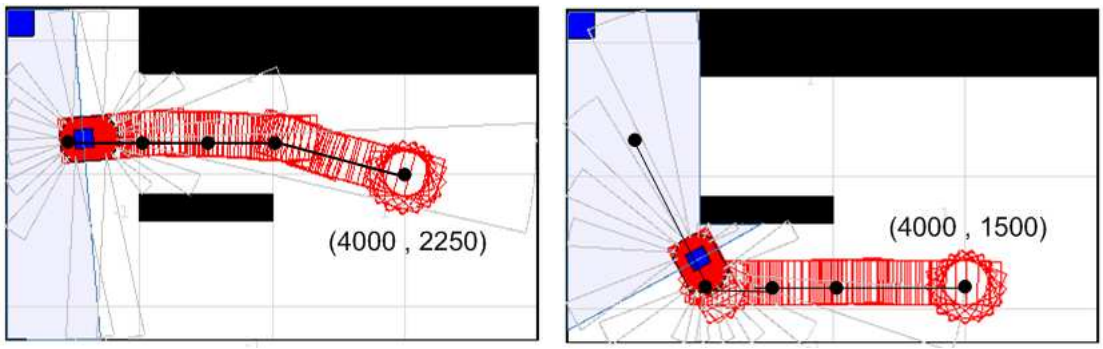
Figure 5.10 – Environnement de test simplifié

TABLEAU 5.8 – Fichier de définition utilisé pour la preuve de fonctionnement

```
FREESPACE 4 // number of spaces where the robot can go
1000 1000 0 1000 3500 0 2000 3500 0 2000 1000 0
2000 1000 0 2000 1900 0 3000 1900 0 3000 1000 0
3000 1000 0 3000 3000 0 5000 3000 0 5000 1000 0
2000 2100 0 2000 3000 0 3000 3000 0 3000 2100 0
```

Les points noirs représentent les différents points du trajet à suivre. Les deux trajets passent par une cellule intermédiaire et le centre de celle-ci devient un objectif. Des points de transitions sont ajoutés aux frontières entre les cellules. L'objectif est de permettre d'optimiser les trajets. Ce qui est particulièrement utile pour permettre au robot de passer dans les passages étroits. La position de ces points est donnée dans le fichier de définition de l'environnement d'opération. Il faut remarquer que dans les deux tests, le robot est en mesure de définir un trajet en fonction de la position de départ et de le suivre. Dans le deuxième cas, le robot est bloqué avant de pouvoir terminer le dernier segment. La raison est simple, la trajectoire en ligne droite fait passer le robot trop près d'un obstacle et il se trouve bloqué.

Les résulats des tests permettent de dire que le robot est en mesure de définir un trajet en fonction de la position de départ et de le suivre. Dans l'un de tests, le robot est bloqué avant de pouvoir terminer le dernier segment. La raison est simple, la trajectoire en ligne droite fait passer le robot trop près d'un obstacle et il se trouve bloqué par le comportement *Avoid*



(a) Trajet complet (b) Trajet où le robot est bloqué

Figure 5.11 – Exemples de suivi de trajectoires en mode autonome

Front. Ce comportement est relié au fait que le comportement ralentit le robot en fonction de la proximité de l'obstacle et qu'il ne cherche pas à donner une commande moteur qui permettrait au robot de passer à côté de l'obstacle. Il faut rappeler que le choix de la mise en œuvre de l'algorithme d'évitement d'obstacle a été fait de manière à ne pas biaiser les résultats des tests de téléopération.

CONCLUSION

Le présent ouvrage explique la mise en place des aspects électroniques et décisionnels du premier prototype de plate-forme de téléprésence baptisée Télérobot. L'accent est mis sur l'intégration des composants nécessaires pour mettre en œuvre une première itération des fonctionnalités souhaitées sur la plate-forme. L'interface électrique et électronique du Télérobot avec sa base de locomotion est fonctionnelle et caractérisée. Les performances des capteurs de proximité nous permet d'établir un comportement d'évitement d'obstacle simple, ralentissant en fonction de la proximité des obstacles. Les capacités en localisation par odométrie, *NorthStar* et CARMEN démontre sa supériorité par rapport aux autres approches avec des performances beaucoup plus stables qu'avec les autres méthodes. Télérobot est en mesure de se déplacer dans des environnements réels de manière autonome, générant des trajectoires dans des environnements connus, exécutant les commandes d'un utilisateur à distance et s'arrêtant devant les obstacles et sans tomber dans des trous. L'autonomie énergétique démontrée du robot est de trois heures, ce qui répond aux exigences fixées.

De juillet à septembre 2006, le robot fut utilisé de façon satisfaisante dans des tests exhaustifs portant sur l'interface-utilisateur, avec plus de trente participants et sur une période de six semaines, sans démontrer de défaillance aux niveau électronique ou décisionnel.

Parmi les limitations actuelles du prototype, les deux principales portent sur l'évitement d'obstacle avec reprise de trajectoire et la recharge autonome. Il est possible de voir à la figure 5.11 que le comportement d'évitement d'obstacle actuel ne permet pas au robot de se rendre à son objectif quand il y a un obstacle dans son chemin. Le robot va rester sur place jusqu'à ce que l'opérateur intervienne pour le dégager. L'utilisation de comportement d'évitement d'obstacle plus évolué permettrait de solutionner ce problème. La recharge autonome serait nécessaire pour permettre au robot d'opérer pour de longues périodes dans les domiciles. Le bloc *Survive* permet d'identifier le besoin de recharge à partir du niveau de tension des batteries. Il reste à lancer la génération d'un trajet qui a pour objectif une station de recharge dont la position est connue par le robot. Un comportement guidant le

robot pour le branchement à la station de recharge devrait être conçu. Un example a déjà été mis au point au LABORIUS [17]. Par contre, il faudra faire des modifications au niveau des chargeurs ainsi qu'au niveau de la carte d'alimentation du robot pour contrôler les différentes étapes pour recharger les batteries de manière adéquate (e.g., modifier le courant utilisé pour recharger les batteries en fonction de la tension de celles-ci). L'autre problème est de développer un système mécanique pour faire le branchement physique entre le robot et la station.

Le prochaine étape dans la poursuite de ce projet est d'amener le robot dans les domiciles et de valider ses capacités de navigation autonome dans des conditions d'opération hors laboratoire. L'architecture décisionnelle pourra être reprise telle qu'elle, avec des ajustements au niveau du module de localisation (e.g., utiliser la plus récente version de CARMEN) et d'évitement d'obstacles (e.g., pour le contournement d'obstacles en suivant une trajectoire). L'utilisation d'un capteur laser URG04 d'Hokuyo devrait aussi augmenter l'autonomie énergétique du robot. La validation du prototype dans l'usage souhaité, soit de réaliser une téléprésence en transportant un vidéophone, pourra alors être complétée afin de passer à la réalisation dans un contexte réel d'usage pour les soins aux personnes âgées.

Références

[1] R. Arkin. *Behavior-Based Robotics*. The MIT Press, 1998.

[2] R. Betke et L. Gurvits. Mobile robot localization using landmarks. *IEEE Transactions on Robotics and Automation*, pages 251–263, 1997.

[3] J. Bisson, F. Michaud, et D. Létourneau. Relative positioning of mobile robots using ultrasounds. *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1783–1788, 2003.

[4] P. Boissy, H. Corriveau, F. Michaud, M.P. Royer, et D. Labonté. Exploring the potential use of home mobile telepresence for telehomecare : a qualitative study with healthcare professionals and community living elderly. *Journal of Telemedecine and Telecare*, 2006.

[5] D. Burschka et G. D. Hager. V-GPS(SLAM) : Vision-based inertial system for mobile robot. *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, pages 409–415, 2004.

[6] H. Choset, E. Acar, A. Rizzi, et J. Luntz. Exact cellular decompositions in terms of critical points of morse functions. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2270–2277, 2000.

[7] H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, et S. Thrun. *Principles of Robot Motion*. MIT Press, 2005.

[8] C. Côté, Y. Brosseau, D. Létourneau, C. Raïevsky, et F. Michaud. Using marie in software development and integration for autonomous mobile robotics. *International Journal of Advanced Robotic Systems, Special Issue on Software Development anf integration in Robotics*, 3(1) :55–60, 2006.

[9] C. Côté, D. Létourneau, F. Michaud, J.-M. Valin, Y. Brosseau, C. Raïevsky, M. Lemay, et V. Tran. Code reusability tools for programming mobiel robots. *Proceedings IEEE/RSJ International Conference on Intellignet Robots and Systems*, pages 1820–1825, 2004.

[10] G. Dudek et M. Jenkin. *Computational Principles of Mobile Robotics*. Cambridge University Press, 2000.

[11] Y. Hada et S. Yuta. The second stage experiments on long term activity of autonomous mobile robots – repetitive navigation for one week in a corridor. *Journal of Robotics and Mechatronics*, 14(4) :375–381, 2002.

[12] M. Hans, B. Graf, et R. Schraft. Robotics home assistant care-o-bot : Past – present – future. *Proceedings IEEE International Workshop on Robot and Human Interactive Communication ROMAN'02*, pages 380–385, 2002.

[13] D. Hähnel, W. Burgard, D. Fox, K. Fishkin, et M. Philipose. Mapping and localization with RFID technology. *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, pages 1015–1020, April 2004.

[14] J. Kosecka et F. Li. Vision based topological markov localization. *Proceedings of the 2004 IEEE International Conference on Robotics & Automation*, pages 1481–1486, 2004.

[15] P. MacKenzie et G. Dudek. Precise positionning using model-based maps. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1615–1621, 1994.

[16] F. Michaud, D. Létourneau, D. Lepage, Y. Morin, F. Gagnon, P. Giguère, É. Beaudry, Y. Brosseau, C. Côté, A. Duquette, J.-F. Laplante, M.-A. Legault, P. Moisan, A. Ponchon, C. Raïevsky, M.-A. Roux, T. Salter, J.-M. Valin, S. Caron, P. Masson, F. Kabanza, et M. Lauria. A brochette of socially interactive robots. *Proceedings American Association for Artificial Intelligence Conference*, pages 1733–1734, 2005.

[17] F. Michaud et E. Robichaud. Sharing charging stations for long-term activity of autonomous robots. *Proceedings IEEE/RSJ International Conference on Intellignet Robots and Systems*, pages 2748–2751, 2002.

[18] M. Montemerlo, N. Roy, et S. Thrun. Perspectives on standardization in mobile robot programming : The carnegie mellon navigation (carmen) toolkit. *Proceedings of the Conference on Intelligent Robots and Systems (IROS)*, pages 2436–2441, 2003.

[19] I. Nourbakhsh, Ju, Bobenage, S. Grange, R. Lutz, R. Meyer, et A. Soto. An affective mobile robot educator with a full-time job. *Artificial Intelligence*, 14(1–2) :95–124, 1999.

[20] M. Pollack. Pearl : Mobile robotic assistant for the elderly. *AAAI Workshop on Automation as Eldercare*, 2002.

[21] B. Nissanka Priyantha, Anit. Chakraborty, et H. Balakrishnan. The cricket location-support system. *Proceedings of the Sixth Annual ACM International Conference on Mobile Computing and Networking (MOBICOM)*, pages 32–43, 2000.

[22] N. Roy, G. Baltus, D. Fox, F. Gemperle, J. Goetz, T. Hirsch, D. Magaritis, M. Montemerlo, J. Pineau, J. Schulte, et S. Thrun. Towards personal service robots for the elderly. *Proceedings of the Workshop on Interactive Robotics and Entertainment (WIRE)*, 2000.

[23] R. Siegwart et I. Nourbakhsh. *Introduction to Autonomous Mobile Robots*. The MIT Press, 2004.

[24] R. Sim et G. Dudek. Mobile robot localization from learned landmark. *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1998.

[25] S. Thrun. Learning metric-topological maps for indoor mobile robot and navigation. *Artificial Intelligence*, 99(1) :21–71, 1998.

[26] R.T. Vaughan, B.P. Gerkey, et A. Howard. On device abstraction for portable, reusable robot code. *Proceedings IEEE/RSJ International Conference on Intellignet Robots and Systems*, pages 2421–2427, 2003.